BELIEF MAINTENANCE UNDER RETRACTION IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Retracting an inferred belief requires withdrawing its support while reassessing the conclusions that depend on it. Such changes are nonmonotonic (NMR): removing one justification can invalidate a conclusion, restore an alternative, or enable a new inference. We investigate how large language models (LLMs) perform this belief maintenance by formalizing two procedures: EDITING an existing reasoning state and REBUILDING it from revised information. Within a temporal argumentation framework, we compare these procedures under shared counterfactuals and support cuts. We establish conditions for agreement on belief values and show that identical current values can conceal differences in support histories that emerge under subsequent retraction. Using NMR-Retraction, we evaluate whether LLMs perform support withdrawal and belief revision or preservation in accordance with the semantics of each procedure. Experiments reveal substantial difficulties in downstream retraction and history-dependent revision, despite stronger performance in accepting counterfactual assumptions. Supervised fine-tuning improves performance on unseen graphs, showing that these maintenance capabilities can be strengthened through training. Together, these findings highlight the importance of procedural knowledge for reliable belief maintenance in LLM reasoning.

1 INTRODUCTION

As large language models (LLMs) reason from incomplete information, new evidence can weaken or overturn earlier conclusions (Rudinger et al., 2020; Allaway & McKeown, 2025). Revising a conclusion can affect beliefs that depend on it (Cohen et al., 2024). Reliable reasoning therefore requires **belief maintenance**: determining which conclusions to withdraw, revise, or preserve as their support changes (Doyle, 1979). Accepting new information is insufficient if subsequent reasoning relies on conclusions that are no longer justified. Evaluations of belief revision in LLMs highlight the need to understand how models revise conclusions while preserving justified ones (Wilie et al., 2024).

To examine these decisions, we evaluate LLMs against specified maintenance semantics. An update alone may not determine how other beliefs should change, making explicit revision criteria essential for evaluation (Hase et al., 2024). Here, these semantics govern support withdrawal, propagation, and the role of retained inference histories in retraction. For example, a conclusion may have multiple justifications, so losing one need not require withdrawing it (de Kleer, 1986).

We focus on two approaches to belief maintenance in LLM reasoning: EDITING and REBUILDING. Editing revises existing conclusions and their supporting inferences, whereas rebuilding derives conclusions anew from the initial facts and rules under a changed assumption. This distinction parallels incremental maintenance and recomputation in deductive systems (Motik et al., 2015). For LLMs, the choice concerns how prior reasoning is used when maintaining persistent beliefs (Kassner et al., 2021) and drawing inferences from revised knowledge (Zhong et al., 2023).

To compare these procedures, we apply **counterfactual interventions** that reverse a target proposition's current truth value. Both EDIT and REBUILD receive the same new assumption and exclude the same selected rules to interrupt existing derivations of the target's previous value, following kernel-based support withdrawal (Hansson, 1994). We formalize their execution in a **temporal logical argumentation framework** that records current beliefs and the order of rule applications. Each belief takes the value of its most recent valid application, so withdrawing support can make an earlier application decisive. Figure 1 illustrates how a downstream conclusion remains true even

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

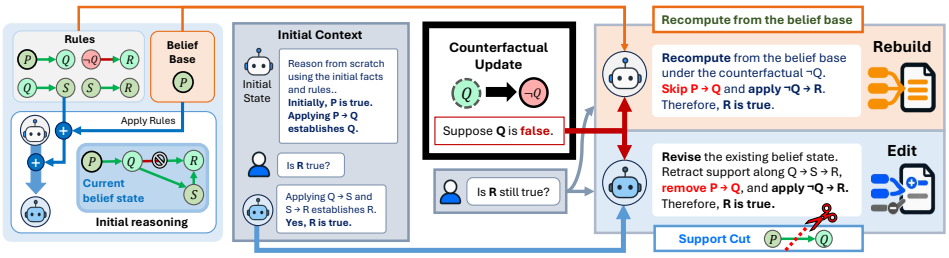


Figure 1: **Editing and rebuilding under counterfactual retraction.** Rebuilding reasons anew from the initial belief base, whereas editing maintains the existing state through support retraction and renewed inference. Both can preserve the same conclusion while changing its justification.

as its original derivation loses support and the new assumption enables a different derivation. We establish sufficient conditions for editing and rebuilding to agree on belief values within regions of the reasoning graph, and analyze how the location of support withdrawal affects which intermediate beliefs are retained. We also show that states with identical current beliefs can respond differently to the same later retraction because their support histories differ. This connects our analysis to the distinction between current beliefs and richer epistemic states in revision (Darwiche & Pearl, 1997).

We introduce NMR-RETRACTION to evaluate required revisions and justified preservation under each maintenance procedure. Following benchmarks with formal semantics (Tafjord et al., 2021; Xiu et al., 2022), we compute symbolic reference answers for beliefs that support the target and conclusions derived from it. Models do not consistently accept the counterfactual assumption and are less accurate, on average, at predicting the effects of retraction on related beliefs. Further analyses reveal difficulties in using retained inference histories to determine which beliefs to revise or preserve. Supervised fine-tuning improves maintenance performance, suggesting that training can strengthen the procedural knowledge needed to manage beliefs as their support changes.

Together, our formal analysis and experiments connect explicit maintenance semantics with the revisions and preservations LLMs are expected to perform. This allows us to distinguish differences between the procedures' prescribed outcomes from errors in executing them (Hase et al., 2024). The observed failures and training gains point to the importance of procedural knowledge for withdrawing support, propagating changes, and preserving justified conclusions.

2 RELATED WORK

Belief revision and knowledge maintenance in LLMs. Research on defeasible inference examines how new evidence changes the strength of existing conclusions (Rudinger et al., 2020; Allaway & McKeown, 2025). This motivates evaluating both necessary revision and justified preservation as evidence, facts, or rules change (Wilie et al., 2024; Helwe et al., 2025). When rules conflict, accepted conclusions also depend on preferences or the choice between skeptical and credulous reasoning (Kazemi et al., 2023; Xiu & Liu, 2025). A related concern arises in knowledge editing: changing a fact should be reflected in the conclusions that depend on it, yet successful edits do not ensure such consistency (Zhong et al., 2023; Cohen et al., 2024).

Determining which beliefs should change therefore requires explicit epistemic criteria, as emphasized by Hase et al. (2024). For persistent beliefs, symbolic memory and dependency tracking provide mechanisms for enforcing consistency and propagating retractions (Kassner et al., 2021; He et al., 2026). Nevertheless, evaluations of conversational memory show that outdated assumptions can persist in subsequent responses despite updated evidence (Chao et al., 2026). Motivated by this gap, we evaluate whether LLMs revise and preserve beliefs as required by explicit procedures for withdrawing support and managing inference histories. Appendix A further clarifies how these procedures differ from prior approaches.

Belief contraction and support maintenance. Classical belief revision characterizes rational contraction and revision of belief sets (Alchourrón et al., 1985). For belief bases, kernel contraction formalizes support withdrawal through incisions that intersect every minimal subset entailing the rejected belief (Hansson, 1994), with relevance requirements constraining which removals are justified

(Ribeiro, 2022). Because losing one derivation need not invalidate a conclusion, truth maintenance systems track justifications and alternative assumption environments (Doyle, 1979; de Kleer, 1986). The same concern motivates incremental view and Datalog maintenance, where deletion must account for conclusions that remain derivable (Gupta et al., 1993; Motik et al., 2015). Provenance semirings provide an algebraic representation of how derived results depend on their input facts (Green et al., 2007). Across successive changes, maintaining current conclusions also requires considering the state that governs future updates. Iterated belief revision distinguishes accepted beliefs from richer epistemic states (Darwiche & Pearl, 1997), while dynamic logic programming formalizes sequences of rule updates (Alferes et al., 1999). Within such sequences, causal rejection resolves conflicts by allowing newer applicable rules to override older ones (Eiter et al., 2002). We examine how retaining or rebuilding ordered application histories affects belief maintenance under the same support cut.

3 Belief Retraction Framework

A proposition may receive competing values from different rule applications. When assumptions change, the application governing its current value may become invalid, while alternative support remains available (Doyle, 1979; de Kleer, 1986). We formalize these dynamics in a temporal argumentation framework that records ordered rule applications alongside current belief values.

In structured argumentation, preferences help determine which conflicting arguments prevail (Modgil & Prakken, 2013). Here, we adopt recency-based priority:[1] each proposition takes the value assigned by its most recent valid application. Withdrawing that application can therefore make an earlier valid application decisive again. We first define proposition-level belief histories and their resolution, then introduce inference, counterfactual retraction, and two strategies for maintaining the resulting belief state. Alternative resolution semantics and the rationale for our choice are discussed in Appendix A.

3.1 Proposition-Level Belief Histories

Let $\mathcal{P}$ be a finite set of propositions and $\mathcal{L} = \{p, \neg p \mid p \in \mathcal{P}\}$ the corresponding set of literals. A finite rule set $\mathcal{R}$ consists of rules

$$r_i : \alpha_i \to \beta_i, \qquad \alpha_i \in \mathcal{L} \cup \{\varnothing\}, \qquad \beta_i \in \mathcal{L}, \tag{1}$$

where $\mathrm{prem}(r_i) = \alpha_i$ and $\mathrm{concl}(r_i) = \beta_i$. A nonempty premise represents an inference from one literal to another. An empty-premise rule represents a directly given value: $\varnothing \to p$ directly supports p, whereas $\varnothing \to \neg p$ directly supports $\neg p$. Directly given and inferred values can therefore be represented uniformly as rule applications.

Following justification-based truth maintenance (Doyle, 1979), we retain the justifications alongside belief values. In our framework, these justifications are recorded as ordered rule applications. At step t, applications concluding either p or $\neg p$ are stored in the ordered justification history

$$H_t(p) = \langle (r_{i_1}, v_1), \ldots, (r_{i_m}, v_m) \rangle, \quad v_j = \mathrm{val}_t(p, j) \in \{0, 1\}, \quad \mathrm{atom}(\mathrm{concl}(r_{i_j})) = p. \tag{2}$$

The order records when the applications were produced, from earlier to later, and each new application is appended to the right. The value $v_j = 1$ means that the application at position j remains supported, whereas $v_j = 0$ means that its supporting condition has been invalidated. Invalid applications remain in their original positions. If the same rule becomes supported again after being invalidated, a new application record is appended rather than reactivating the earlier record.

Let $H_t^{\mathrm{valid}}(p)$ contain the valid rules in their original order. The governing rule is its last element:

$$H_t^{\mathrm{valid}}(p) = \left\langle r_{i_j} \;\middle|\; \mathrm{val}_t(p, j) = 1 \right\rangle, \qquad r_t^*(p) = \begin{cases} \mathrm{last}(H_t^{\mathrm{valid}}(p)), & H_t^{\mathrm{valid}}(p) \neq \langle\rangle, \\ \bot, & H_t^{\mathrm{valid}}(p) = \langle\rangle. \end{cases} \tag{3}$$

The current value of p is resolved from the conclusion of this rule:

$$\mathrm{resolve}_t(p) = \begin{cases} \mathsf{t}, & r_t^*(p) \neq \bot \text{ and } \mathrm{concl}(r_t^*(p)) = p, \\ \mathsf{f}, & r_t^*(p) \neq \bot \text{ and } \mathrm{concl}(r_t^*(p)) = \neg p, \\ \mathsf{u}, & r_t^*(p) = \bot. \end{cases} \tag{4}$$

[1] More generally, a resolution function determines the belief value from available positive and negative support. We choose recency under the assumption that newer information takes precedence in LLM reasoning.

Thus, the resolved value records only the conclusion of the last valid rule, while the history retains the earlier justifications that may govern after a later retraction. Since validity is already stored in each history, we write the complete historical belief state compactly as $X_t = (H_t(p))_{p\in\mathcal{P}}$.

Illustrative example. Consider $r_0 : \varnothing \to p$, $r_1 : a \to \neg p$, and $r_2 : b \to p$, applied in this order. When all three applications are valid, r_2 governs p. If the support for b is later withdrawn, the application of r_2 becomes invalid and r_1 becomes the last valid rule:

$$\begin{aligned} H_t(p) &= \langle (r_0, 1), (r_1, 1), (r_2, 1) \rangle , \qquad \mathrm{resolve}_t(p) = \mathtt{t}, \\ H_{t+1}(p) &= \langle (r_0, 1), (r_1, 1), (r_2, 0) \rangle , \quad \mathrm{resolve}_{t+1}(p) = \mathtt{f}. \end{aligned} \tag{5}$$

The history is preserved across the update; only the validity of the withdrawn application changes. The resolved value therefore falls back from the conclusion of r_2 to that of r_1 without reconstructing the history from scratch.

3.2 Multiplicity of Support Retraction

Retraction requires choosing which support to withdraw. The same target can be retracted in different ways, with different effects on the remaining beliefs. Consider a support path

$$\pi_{\alpha\rightsquigarrow\beta} = \langle \ell_0 = \alpha, r_1, \ell_1, \ldots, r_k, \ell_k = \beta \rangle, \tag{6}$$

with fixed initial support for α. One may remove a single rule $\{r_i\}$, $1 \le i \le k$, or retract an intermediate literal ℓ_i by removing its upstream support $\{r_1, \ldots, r_i\}$, $1 \le i < k$.

Proposition 1 (Multiplicity of support retraction)**.** *For an initially supported, isolated chain with $k \ge 2$ and no other support, including historical fallback, these two families contain $2k-2$ distinct cuts. All retract β. Applied separately to the same initial state, they yield k distinct current-value outcomes, with $k-1$ cuts sharing one outcome.*

Proof sketch. Only $\{r_1\}$ occurs in both families. All literal cuts remove r_1, leaving only α supported, while rule cuts $\{r_2\}, \ldots, \{r_k\}$ preserve different prefixes.

With multiple support paths, a valid choice must cut every current path. We express this constraint by adapting the incision principle of kernel contraction (Hansson, 1994):

$$C \cap \mathcal{R}(\pi) \neq \varnothing \qquad \text{for every current support path } \pi, \tag{7}$$

where $\mathcal{R}(\pi)$ includes its empty-premise support rule. This condition concerns the pre-retraction state; fallback may enable alternative support. It does not generally determine a unique cut. We therefore compare editing and rebuilding under the same C, separating support selection from subsequent maintenance. Related results across cut depths appear in Section 5.1.

4 Semantic Equivalence of Rebuild and Edit

Inference. Inference propagates resolved values through rules in a fixed topological order. Empty-premise rules apply directly; a premise p requires $\mathrm{resolve}(p) = \mathtt{t}$, while $\neg p$ requires $\mathrm{resolve}(p) = \mathtt{f}$. Each newly supported rule appends a valid application record to its conclusion's history, and the resulting resolved value is used in subsequent inference. A continuously supported application retains its record unless renewed as CF-grounded support. Renewal after invalidation or a transition from ordinary to CF-grounded support appends a new application record. We write $\mathrm{Infer}(X)$ for the closure reached when no further updates occur.

Counterfactual retraction. For a target q, a counterfactual imposes $c \in \{\mathtt{t}, \mathtt{f}\}$ with $c \neq \mathrm{resolve}_t(q)$. Let $\ell(q, c)$ be q if $c = \mathtt{t}$ and $\neg q$ otherwise. The intervention introduces an empty-premise rule and appends its application to the target's history, where $\|$ denotes sequence concatenation:

$$r^{\Delta}_{q\leftarrow c} : \varnothing \to \ell(q, c), \qquad H^{\Delta}_t(q) = H_t(q) \,\|\, \langle (r^{\Delta}_{q\leftarrow c}, 1) \rangle . \tag{8}$$

We denote the state after insertion by $X_t \oplus \Delta$. In Edit, insertion follows forward and backward retraction; in Rebuild, the counterfactual is included in the revised initial state. The intervention receives the highest priority at q. Counterfactual priority is realized through application order: CF-induced applications are processed in a final propagation pass under the fixed dependency and rule order, so that their valid records occur after competing ordinary applications at closure. The same last-valid-application rule therefore determines the resolved value. This convention applies to both procedures; we denote inference under it by Infer_{Δ}. Insertion changes the target's resolved value but does not itself invalidate existing applications or propagate downstream consequences.

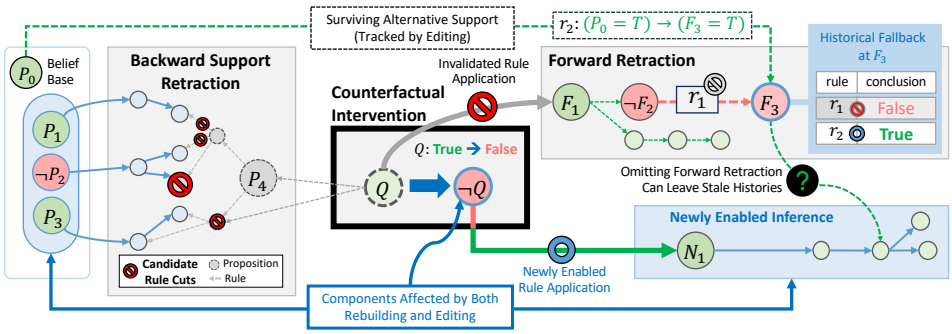


Figure 2: **Forward retraction, backward retraction, and inference.** In preparation for changing Q from true to false, forward retraction invalidates unsupported applications along the previous downstream path. Invalidating the application of r_1 allows the surviving application of r_2 to govern F_3 through historical fallback. Backward retraction removes selected rules supporting the previous value of Q. Subsequent inference propagates the counterfactual through N_1, with conclusions along this newly enabled path potentially affected by the fallback at F_3.

4.1 Retraction Maintenance

Retraction maintenance reassesses existing conclusions, withdraws support for the target's previous value, and then applies and propagates the counterfactual. Figure 2 illustrates how these components interact through support dependencies and historical fallback.

- **Backward retraction** ($\mathrm{BRet}_{q,C}$) traces support for the target q's previous value against the dependency direction, removes the selected rules C from the active rule set, and invalidates their existing applications.
- **Forward retraction** (FRet_q) retracts consequences supported by q's previous value, following outgoing dependencies in the DAG to invalidate applications whose premises no longer match the resolved values.
- **Inference** (Infer) applies newly supported rules in the fixed topological and rule order, appending valid application records and propagating the resolved values.

During either retraction phase, invalidated records retain their original positions, with validity changed from 1 to 0. Invalidation may expose an earlier valid application through historical fallback. Any resulting value change propagates along outgoing dependencies, invalidating incompatible applications and potentially triggering further fallback. Newly enabled rule applications are deferred to inference. During retraction, q is temporarily excluded as a premise source so that its previous value cannot regenerate the withdrawn consequences. Our editing procedure applies FRet_q followed by $\mathrm{BRet}_{q,C}$, using the cut C selected from the pre-retraction state. It then inserts the counterfactual $q = c$ and restores q as a premise source, obtaining

$$\widetilde{X}_t^{\Delta,C} = \mathrm{BRet}_{q,C}\left(\mathrm{FRet}_q(X_t)\right) \oplus \Delta. \tag{9}$$

Inference under Infer_Δ first proceeds from the affected upstream propositions in the fixed topological and rule order, propagating fallback values and newly enabled ordinary applications. The protected counterfactual $q = c$ is then propagated in a final pass under the same order, with counterfactual-induced applications taking priority over competing ordinary applications. Maintenance continues until no further updates occur.

4.2 Rebuilding and Editing

Rebuilding and editing share the counterfactual Δ and support cut C, but differ in how they maintain histories. Let X_{init} contain the initial empty-premise applications, and let $\widetilde{X}_{\mathrm{init}}^{\Delta,C}$ incorporate Δ while excluding C. Rebuilding reconstructs histories from this state. Editing instead uses the existing state after retraction and counterfactual insertion:

$$\mathrm{Rebuild}_{\Delta,C}(X_t) = \mathrm{Infer}_\Delta\left(\widetilde{X}_{\mathrm{init}}^{\Delta,C}\right), \qquad \mathrm{Edit}_{\Delta,C}(X_t) = \mathrm{Infer}_\Delta\left(\widetilde{X}_t^{\Delta,C}\right). \tag{10}$$

Under shared counterfactual priority and propagation order, the procedures agree where common counterfactual-induced applications govern. Elsewhere, retaining and reconstructing histories may select different governing applications. Even when current values agree, different surviving histories can yield different outcomes under later retractions (Darwiche & Pearl, 1997).

A counterfactual fixes the target's new value, but its effects on other beliefs depend on support withdrawal, propagation, and surviving histories. We characterize retraction depths, regions where editing and rebuilding agree on current values, and how different histories can yield different outcomes under later retractions. Full proofs appear in Appendix B.2.

4.3 Regional Value Equivalence

Rebuilding constructs fresh histories, while editing retains surviving applications and appends newly enabled ones. Recomputing the same propositions therefore need not produce the same application order or resolved values. We define current-value equivalence as

$$X \equiv_{\mathrm{val}} Y \iff \mathrm{resolve}_X(p) = \mathrm{resolve}_Y(p) \quad \forall p \in \mathcal{P}. \tag{11}$$

We distinguish three regions. The *upstream region* $\mathcal{U}_C$ contains propositions that precede both the target q and every cut location in the topological order, where a cut location is the head proposition of a removed rule. The *contraction-only region* $\mathcal{B}_C$ contains propositions whose edited histories change only through invalidation of cut-rule applications. Their uncut incoming rules retain their original applicability in both procedures, without additional renewal caused by intermediate value changes. The *On-CF-Path region* $\mathcal{F}_\Delta$ contains q and propositions supported at closure by valid applications of the common counterfactual propagation. Propositions outside $\mathcal{F}_\Delta$ are *Off-CF-Path*.

Proposition 2 (Regional value equivalence)**.** *Suppose X_t is constructed from the initial belief base under the fixed inference order. Let $E = \mathrm{Edit}_{\Delta,C}(X_t)$ and $R = \mathrm{Rebuild}_{\Delta,C}(X_t)$ use the same cuts, counterfactual, surviving rules, and inference order, and terminate in closed states. If their common CF-induced applications have final priority over competing ordinary applications at closure, then*

$$\mathrm{resolve}_E(p) = \mathrm{resolve}_R(p) \quad \forall p \in \mathcal{U}_C \cup \mathcal{B}_C \cup \mathcal{F}_\Delta. \tag{12}$$

Proof sketch. In $\mathcal{U}_C$, neither the cuts nor the counterfactual affect the incoming rules and premises, so both procedures reproduce the initial values. In $\mathcal{B}_C$, both omit the cut-rule applications and preserve the relative order of the remaining valid rules. In $\mathcal{F}_\Delta$, the common propagation produces the same ordered CF-induced applications, whose validity and final priority determine the same values. Related empirical results appear in Section 5.2.

4.4 Hidden Histories and Sequential Divergence

Iterated revision and contraction distinguish current beliefs from the epistemic state governing subsequent changes (Darwiche & Pearl, 1997; Konieczny & Pino Pérez, 2017). In our setting, even when all current values agree, a later retraction may expose differently ordered surviving support and produce different values. For a subsequent update $\Gamma = (\Delta_\Gamma, C_\Gamma)$, we write $\mathrm{Ret}_\Gamma = \mathrm{Edit}_{\Delta_\Gamma, C_\Gamma}$ for editing under that update.

Theorem 1 (Value equivalence is not sequential equivalence)**.** *There exist historical states X, Y over the same rule set and a common subsequent retraction Γ such that*

$$X \equiv_{\mathrm{val}} Y, \qquad \mathrm{Ret}_\Gamma(X) \not\equiv_{\mathrm{val}} \mathrm{Ret}_\Gamma(Y). \tag{13}$$

Proof sketch. Let a common latest ordinary application govern p in both states, with two earlier ordinary applications supporting opposite values in reversed orders. Choose all other current values to agree. A common retraction invalidates the governing application while leaving the earlier applications continuously supported and enabling no new application at p. Their order is therefore preserved, and the resulting fallback values are opposite.

The relevant hidden information is surviving valid support and its order; invalid records alone do not establish sequential divergence. The theorem shows that current-value agreement does not determine how a historical state responds to later withdrawal. Its construction and implications for retained histories and computational cost appear in Appendix B.2. In Section 5.3, we examine a related single-update maintenance task using Off-CF-Path queries with different Edit and Rebuild references. The rationale and scope of this evaluation are discussed in Appendix D.4.

5 EXPERIMENTS

Dataset. NMR-RETRACTION comprises six-layer graphs (L_0–L_5) with 18–30 propositions and signed unary rules between adjacent layers. A counterfactual at L_3 enables evaluation of upstream support retraction (L_1–L_2) and downstream maintenance (L_4–L_5). We computed belief values, support histories, and procedure-specific reference answers. Procedure comparisons use 900 training graphs (2,907 queries) for tuning and 1,800 test graphs (5,755 queries). The RETRACTION TEST evaluates backward and forward retraction on 360 graphs (4,115 queries). See Appendix C for details.

Models and evaluation. We evaluate six models from the Qwen3 (Yang et al., 2025) and Gemma-3 (GemmaTeam et al., 2025) families. Direct prompting requests a belief value, while chain-of-thought (CoT) prompting (Wei et al., 2022) additionally requests a brief supporting explanation. Prompts specify the semantics, intervention, cuts, and computation procedure (Appendix C.3). Main-text results use *initial-only* inputs; results with a supplied belief base or support history appear in Appendix D.7. Llama results and comparisons with supplied initial states are reported separately in Appendix D. We measure accuracy separately on belief maintenance (BM) and belief update (BU) queries, where the reference value matches or differs from the initial value, respectively. Procedure comparisons include REBUILD and EDIT, with a formal comparison in Appendix B.1.

5.1 RETRACTION ACCURACY AND COUNTERFACTUAL PRESERVATION

We first assess whether models preserve the imposed counterfactual value and correctly determine the values of propositions targeted by retraction. Table 1 separates three outcomes. First, preserving the imposed value at L_3 scores 84.0% with Direct and 81.9% with CoT. The CoT decline comes mainly from Gemma3-12B and Gemma3-27B, which sometimes change the imposed value even though backward retraction leaves the counterfactual target unchanged in these instances. Second, retracting invalidated support in L_1–L_2 scores 38.7% with Direct and 68.8% with CoT, showing higher average accuracy with CoT for backward retraction. Third, withdrawing downstream beliefs inferred from L_3 scores 16.0% and 47.1%, respectively, below backward retraction under both prompting methods.

Table 1: Counterfactual acceptance and independent retraction accuracy (%).

	Qwen3-4B		Qwen3-14B		Qwen3-32B		Gemma3-4B		Gemma3-12B		Gemma3-27B		Average	
Query	D	CoT	D	CoT	D	CoT	D	CoT	D	CoT	D	CoT	D	CoT
Counterfactual	94.0	93.6	89.0	99.9	100.0	99.9	41.6	50.6	80.5	67.7	99.2	79.4	**84.0**	81.9
Backward (BU)	5.2	79.0	55.6	79.6	82.8	62.4	43.6	66.1	28.8	64.2	16.3	61.6	38.7	**68.8**
Forward (BU)	6.0	49.0	25.4	61.2	6.6	56.2	19.6	39.5	29.5	31.7	9.0	44.8	16.0	**47.1**

We examine counterfactual preservation at L_3 as backward cuts progressively remove upstream support. The three cumulative depths successively remove supporting rules ($r : L_2 \to L_3$), their antecedents (L_2), and earlier propositions (L_1). Figure 3 shows that, for Gemma3-12B and Gemma3-27B, CoT more often returns U, while Direct more reliably preserves the imposed value. For Gemma3-27B, Direct remains at 98.9–100% across cuts, whereas CoT improves from 57.8% to 77.8% as cuts deepen. These U responses indicate difficulty preserving the imposed value when upstream support is withdrawn. Under the specified semantics, counterfactual priority protects this value at every cut depth, regardless of the remaining ordinary support.

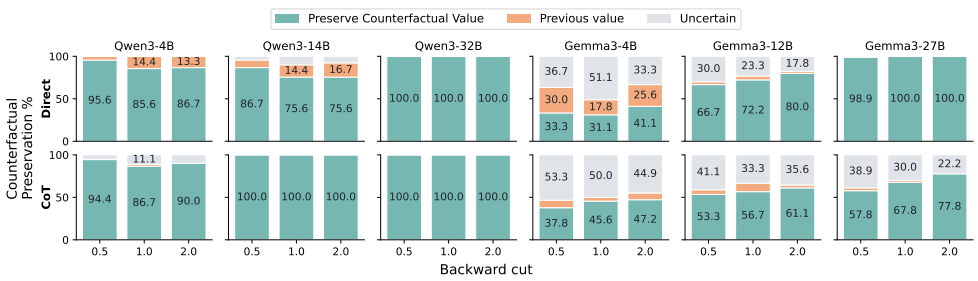


Figure 3: Counterfactual preservation across backward-cut depths.

5.2 REBUILD VS. EDIT ON COUNTERFACTUAL PATHS

Proposition 2 shows that REBUILD and EDIT share reference values along paths from the L_3 counterfactual despite their different state operations. We compare BU and BM accuracy one and two hops downstream (L_4 and L_5) using identical facts, rules, counterfactuals, and cuts. Shared labels make their prediction accuracy directly comparable. Figure 4 shows that direct prompting gives EDIT an average advantage of about four percentage points across BU and BM. The two procedures follow similar patterns across models. CoT raises accuracy at L_4 for both procedures, with some larger models approaching 80%. Their mean accuracies under CoT are close, while the better procedure varies by model. Across BU and BM, mean CoT accuracy falls by about 12 percentage points from L_4 to L_5. Thus, the procedures perform similarly on counterfactual paths, but neither sustains high accuracy over two propagation steps.

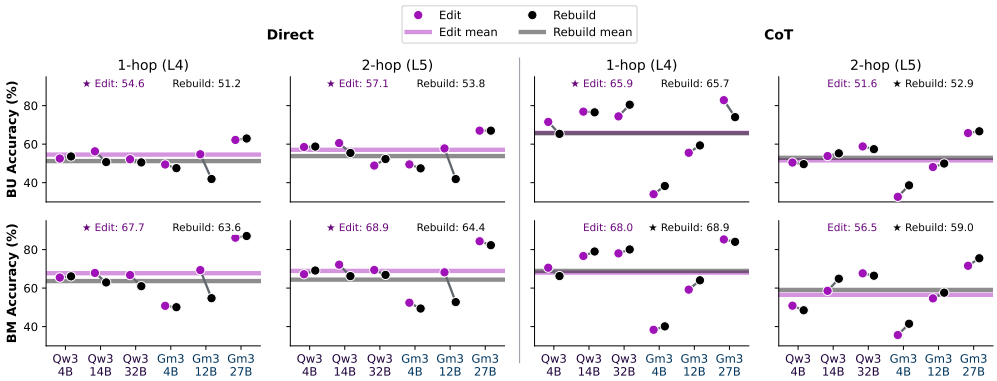


Figure 4: Rebuild vs. Edit: BU/BM accuracy at L_4 and L_5.

5.3 HISTORY-DEPENDENT REVISION OUTSIDE COUNTERFACTUAL PATHS

We evaluate whether models following EDIT correctly revise beliefs outside counterfactual paths, using 82 BU queries without counterfactual-grounded support where EDIT and REBUILD prescribe different values. To assess reassessment of existing histories, we score predictions against EDIT references; REBUILD scores indicate agreement rather than procedure-specific accuracy. Appendix D.4 details this rationale. Figure 5 shows slightly higher mean reference agreement with EDIT instructions under Direct and CoT. Qwen3-14B reaches 53% with EDIT and CoT, but the six-model mean remains below 40%. Models remain inaccurate on these selected revisions even when explicitly instructed to reassess retained histories



Figure 5: Off-CF BU accuracy against EDIT gold label (EDIT ↑, REBUILD ↓).

5.4 FINE-TUNING FOR BELIEF MAINTENANCE

We train six models on 900 graphs with Direct-format EDIT supervision, then test them on 1,800 separate graphs to assess generalization on and off counterfactual paths. Figure 6 shows that after two epochs, Direct accuracy improves in both regions for all six models, with medium and large models reaching similar levels. We also evaluate these Direct-tuned models under CoT. Accuracy improves overall, and some models generate fewer tokens (Figure 7). Additional results, generation examples, and BM/BU analyses appear in Appendix D.5.

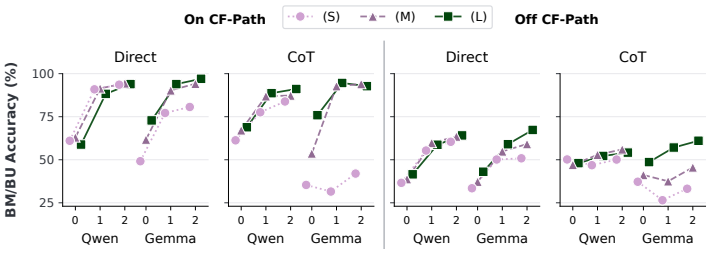


Figure 6: EDIT accuracy across SFT epochs 0–2, prompts, and regions. S/M/L denote Qwen3 4B/14B/32B and Gemma3 4B/12B/27B.

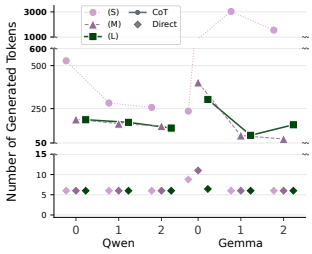


Figure 7: Generated tokens across SFT epochs 0–2.

We also train with GRPO (Shao et al., 2024) for one epoch. It takes longer than SFT but yields average gains below 10%, with some regressions (Appendix D.6). This suggests that direct supervision may be more efficient for learning belief maintenance in this setting.

6 DISCUSSION

Recent work on nonmonotonic reasoning emphasizes specifying how competing conclusions are resolved (Xiu & Liu, 2025). Our analysis examines a complementary question: how beliefs should be maintained as their supporting derivations change. Under our temporal semantics, EDIT and REBUILD can produce different outcomes from the same counterfactual and support cuts. Agreement on current values can also conceal histories that affect later retractions, reflecting the distinction between current beliefs and richer epistemic states (Darwiche & Pearl, 1997). This reinforces calls for explicit epistemic standards (Hase et al., 2024): evaluation should specify both accepted conclusions and state transitions to distinguish differences between procedures from errors in their execution.

The gap between accepting new information and maintaining related beliefs connects our findings to challenges in justified revision (Wilie et al., 2024) and consistent knowledge editing (Cohen et al., 2024). These difficulties persist even when rules, support cuts, and procedures are explicitly supplied. Models are more accurate on counterfactual acceptance than on support retraction and downstream maintenance, although acceptance itself is imperfect. They also struggle with history-dependent outcomes outside counterfactual paths. These results motivate evaluating support withdrawal and downstream reassessment alongside target acceptance. Improvements on unseen graphs after supervised fine-tuning further suggest that explicit training can strengthen these capabilities.

Extending these capabilities to natural-language settings connects to challenges in long-term memory. LongMemEval evaluates knowledge updates across interactions (Wu et al., 2025), while STALE examines adaptation to implicitly outdated memories (Chao et al., 2026). Such settings require identifying which observations support or undermine existing beliefs. Dependency tracking offers a way to propagate the resulting retractions (He et al., 2026). Extending our framework requires representing support from statistical associations, causal relationships, and commonsense knowledge, with priorities informed by context and source reliability. Future work should examine whether maintenance capabilities learned from formal tasks transfer to natural-language updates. Studying how models encode support loss and fallback to surviving justifications may also connect these behavioral evaluations to the internal computations underlying belief change.

7 CONCLUSION

We introduced a framework for nonmonotonic belief maintenance in LLM reasoning based on support dependencies and ordered rule-application histories. We characterized how support cuts affect resulting beliefs, established regional conditions for value equivalence between EDIT and REBUILD, and showed that current agreement need not persist under subsequent retractions. Experiments with NMR-RETRACTION show that LLMs struggle more with maintaining related beliefs than with accepting a new assumption, while supervised fine-tuning improves performance on unseen graphs. Our framework distinguishes semantic choices from execution errors, highlighting the importance of evaluating and learning maintenance procedures for reasoning under changing information.

AI USE STATEMENT

AI tools were used to support the following aspects of this research:

- **Literature exploration.** AI tools assisted in identifying and organizing relevant prior studies and exploring their connections to our research questions, including belief revision, nonmonotonic reasoning, and language-model evaluation.
- **Manuscript preparation.** AI tools assisted with drafting and revising text, improving language and clarity, and organizing the presentation of the theoretical framework, experimental results, and discussion.
- **Mathematical verification.** AI tools assisted in checking mathematical formulations and derivations for errors and consistency with the stated definitions, assumptions, and semantics.
- **Synthetic-data implementation.** AI tools assisted in translating author-provided specifications into data-generation and reference-computation code. The synthetic rules and evaluation criteria were designed by the authors. Reference belief values and support histories were computed by a symbolic engine under the specified semantics.
- **Model inference and training.** AI tools assisted in implementing the experimental pipelines, including vLLM-based inference and fine-tuning code. For the GRPO experiments, they assisted in adapting an existing repository implementation to our tasks and integrating it into the experimental workflow.
- **Implementation verification.** AI tools supported code inspection, debugging, revision, and checks for consistency with the intended specifications. This assistance was used throughout implementation and verification to help identify discrepancies between the code and the author-defined rules and evaluation requirements.

The authors retain responsibility for the research design, implementation and reference computations, interpretation of the results, and all claims and citations in the final manuscript.

ETHICS STATEMENT

This study evaluates language models using synthetic reasoning tasks constructed from abstract propositions and explicit rules. The dataset contains no personal or sensitive information, and the research does not involve human participants. Counterfactual interventions are defined within these synthetic tasks and do not represent claims about real individuals or events. Our evaluation concerns whether models follow specified belief-maintenance semantics, and its findings should be interpreted within this controlled setting.

REFERENCES

Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic*, 50(2):510–530, 1985. doi: 10.2307/2274239. URL https://www.jstor.org/stable/2274239.

José Júlio Alves Alferes, João Alexandre Carvalho Pinheiro Leite, Luís Moniz Pereira, Halina Przymusinska, and Teodor C Przymusinski. Dynamic logic programming. 1999.

Emily Allaway and Kathleen McKeown. Evaluating defeasible reasoning in LLMs with DEFREASING. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 10540–10558, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.529. URL https://aclanthology.org/2025.naacl-long.529/.

Hanxiang Chao, Yihan Bai, Rui Sheng, Tianle Li, and Yushi Sun. Stale: Can llm agents know when their memories are no longer valid?, 2026. URL https://arxiv.org/abs/2605.06527.

Maria Laura Cobo, Diego C Martinez, and Guillermo R Simari. Stable extensions in timed argumentation frameworks. In *International Workshop on Theorie and Applications of Formal Argumentation*, pp. 181–196. Springer, 2011.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024. doi: 10.1162/tacl_a_00644. URL https://aclanthology.org/2024.tacl-1.16/.

Adnan Darwiche and Judea Pearl. On the logic of iterated belief revision. *Artificial Intelligence*, 89(1–2):1–29, 1997. doi: 10.1016/S0004-3702(96)00038-0. URL https://www.sciencedirect.com/science/article/pii/S0004370296000380.

Johan de Kleer. An assumption-based TMS. *Artificial Intelligence*, 28(2):127–162, 1986. doi: 10.1016/0004-3702(86)90080-9. URL https://www.sciencedirect.com/science/article/pii/0004370286900809.

Jon Doyle. A truth maintenance system. *Artificial Intelligence*, 12(3):231–272, 1979. doi: 10.1016/0004-3702(79)90008-0. URL https://www.sciencedirect.com/science/article/pii/0004370279900080.

Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995. doi: 10.1016/0004-3702(94)00041-X. URL https://www.sciencedirect.com/science/article/pii/000437029400041X.

Thomas Eiter, Michael Fink, Giuliana Sabbatini, and Hans Tompits. On properties of update sequences based on causal rejection. *Theory and Practice of Logic programming*, 2(6):711–767, 2002.

GemmaTeam et al. Gemma 3 technical report, 2025. URL https://arxiv.org/abs/2503.19786.

Todd J Green, Grigoris Karvounarakis, and Val Tannen. Provenance semirings. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 31–40, 2007.

Ashish Gupta, Inderpal Singh Mumick, and Venkatramanan Siva Subrahmanian. Maintaining views incrementally. *ACM SIGMOD Record*, 22(2):157–166, 1993.

Sven Ove Hansson. Kernel contraction. *The Journal of Symbolic Logic*, 59(3):845–859, 1994. doi: 10.2307/2275912. URL https://www.jstor.org/stable/2275912.

Peter Hase, Thomas Hofweber, Xiang Zhou, Elias Stengel-Eskin, and Mohit Bansal. Fundamental problems with model editing: How should rational belief revision work in LLMs? *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=LRf19n5Ly3.

Qisong He, Jinwei Hu, Xinmiao Huang, Changshun Wu, Yi Dong, and Xiaowei Huang. Grounded continuation: A linear-time runtime verifier for llm conversations, 2026. URL https://arxiv.org/abs/2605.14175.

Chadi Helwe, Sultan AlRashed, and Francesco Orabona. ReviseQA: A benchmark for belief revision in multi-turn logical reasoning. In *ICML 2025 Workshop on Assessing World Models*, 2025. URL https://openreview.net/forum?id=Z4KBiAYXlI.

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. BeliefBank: Adding memory to a pre-trained language model for a systematic notion of belief. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 8849–8861, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.697. URL https://aclanthology.org/2021.emnlp-main.697/.

Mehran Kazemi, Quan Yuan, Deepti Bhatia, Najoung Kim, Xin Xu, Vaiva Imbrasaite, and Deepak Ramachandran. BoardgameQA: A dataset for natural language reasoning with contradictory information. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=BR1m3JIoKm.

Sébastien Konieczny and Ramón Pino Pérez. On iterated contraction: syntactic characterization, representation theorem and limitations of the levi identity. In *International Conference on Scalable Uncertainty Management*, pp. 348–362. Springer, 2017.

Sanjay Modgil and Henry Prakken. A general account of argumentation with preferences. *Artificial Intelligence*, 195:361–397, 2013. doi: 10.1016/j.artint.2012.10.008. URL https://www.sciencedirect.com/science/article/pii/S0004370212001361.

Boris Motik, Yavor Nenov, Robert Piro, and Ian Horrocks. Incremental update of datalog materialisation: the backward/forward algorithm. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models, 2024. URL https://arxiv.org/abs/2404.15522.

Jandson S. Ribeiro. Kernel contraction and the order of relevance. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 299–308, 2022. doi: 10.24963/kr.2022/30. URL https://proceedings.kr.org/2022/30/.

Rachel Rudinger, Vered Shwartz, Jena D. Hwang, Chandra Bhagavatula, Maxwell Forbes, Ronan Le Bras, Noah A. Smith, and Yejin Choi. Thinking like a skeptic: Defeasible inference in natural language. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4661–4675, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.418. URL https://aclanthology.org/2020.findings-emnlp.418/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Bryan Wilie, Samuel Cahyawijaya, Etsuko Ishii, Junxian He, and Pascale Fung. Belief revision: The adaptability of large language models reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 10480–10496. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.586. URL https://aclanthology.org/2024.emnlp-main.586/.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.10813.

Yeliang Xiu and Yongmei Liu. MultiLogicNMR(er): A benchmark and neural-symbolic framework for non-monotonic reasoning with multiple extensions. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 18372–18405, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.927. URL https://aclanthology.org/2025.emnlp-main.927/.

Yeliang Xiu, Zhanhao Xiao, and Yongmei Liu. LogicNMR: Probing the non-monotonic reasoning ability of pre-trained language models. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 3616–3626, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.265. URL https://aclanthology.org/2022.findings-emnlp.265/.

An Yang et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Zexuan Zhong, Zhengxuan Wu, Christopher Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15686–15702. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.971. URL https://aclanthology.org/2023.emnlp-main.971/.

APPENDIX

CONTENTS

A Semantics of Belief Retraction in Nonmonotonic Reasoning **15**

A.1 Semantic Choices in Prior Work 15

A.2 History and Procedure as Semantic Commitments 15

A.3 Position Relative to LLM Evaluation 16

B Theoretical Analysis and Proofs **17**

B.1 Kernel Contraction and Processing Order 17

B.2 Proofs of Theoretical Results 19

C NMR-Retraction Dataset **23**

C.1 Data Generation 23

C.2 Data Statistics 24

C.3 LLM Prompts 25

D Experiments **28**

D.1 Experimental Settings 28

D.2 Retraction Prediction Accuracy 28

D.3 On CF-Path Evaluation 29

D.4 Off CF-Path Evaluation 30

D.5 Supervised Fine-Tuning 31

D.6 RL Tuning 33

D.7 Inferred versus Supplied Belief States 34

A SEMANTICS OF BELIEF RETRACTION IN NONMONOTONIC REASONING

Belief maintenance requires specifying both accepted conclusions and changes to the underlying state. Our framework makes priorities, application histories, and maintenance procedures explicit to evaluate retraction in LLMs.

A.1 SEMANTIC CHOICES IN PRIOR WORK

Nonmonotonic reasoning admits different choices about how rules apply, how conflicts are resolved, and which conclusions are accepted. Argumentation defines acceptance through attack relations and extension semantics (Dung, 1995), while preferences determine how competing arguments are compared (Modgil & Prakken, 2013). Even with the same extensions, skeptical and credulous reasoning differ in whether conclusions must hold across all extensions or only some, as evaluated by MultiLogicNMR (Xiu & Liu, 2025).

Further choices arise when the reasoning state changes. Kernel contraction specifies the selection of support for removal (Hansson, 1994), while truth maintenance and incremental view maintenance track dependencies and reassess derivations (Doyle, 1979; de Kleer, 1986; Gupta et al., 1993). Iterated revision additionally makes subsequent changes depend on epistemic information beyond current beliefs (Darwiche & Pearl, 1997). Thus, acceptance criteria, support selection, and maintenance policies are distinct commitments that must be specified when comparing reasoning procedures.

Temporal semantics makes additional choices. Causal rejection prioritizes applicable rules from later updates (Eiter et al., 2002), while timed argumentation specifies periods of argument availability (Cobo et al., 2011). Our histories instead record the creation order and continuing validity of individual applications, including renewed applications of previously invalidated rules.

A.2 HISTORY AND PROCEDURE AS SEMANTIC COMMITMENTS

We resolve each proposition using its most recent valid application, subject to counterfactual priority. Recency implements the modeling assumption that newly established support takes precedence. Other deterministic policies can resolve positive and negative support; our analysis uses the specified recency policy.

For $r_+ : a \to q$ and $r_- : b \to \neg q$, with both premises supported and neither application receiving special priority,

$$\begin{aligned} H(q) &= \langle (r_+, 1), (r_-, 1) \rangle, \quad \text{resolve}_H(q) = \mathtt{f}, \\ H'(q) &= \langle (r_-, 1), (r_+, 1) \rangle, \quad \text{resolve}_{H'}(q) = \mathtt{t}. \end{aligned} \tag{14}$$

The same applications remain valid, but their order changes the resolved value. This order is part of the semantics that an implementation must respect.

Maintenance determines how these histories change. Backward retraction removes the selected support rules; forward retraction invalidates applications with unsupported premises. Both phases propagate resulting value changes and may expose surviving support through fallback. Subsequent inference appends newly enabled applications. Continuous support preserves an application's position, whereas renewed support creates a new event. Full editing performs forward retraction, backward retraction, and inference in that order; rebuilding constructs fresh histories under the same counterfactual and cut.

Let $V(X)$ denote the resolved values in state X, and U_Γ a common subsequent maintenance operation. Our analysis identifies states satisfying

$$V(X) = V(Y), \qquad V(U_\Gamma(X)) \neq V(U_\Gamma(Y)) \,. \tag{15}$$

Withdrawing a common governing application can expose differently ordered surviving support (theorem 1). Valid support and its order drive this divergence; invalid records alone do not establish it.

Recency does not guarantee convergence on cyclic graphs. With $\varnothing \to p$ and $p \to \neg p$, repeatedly applying and invalidating the latter rule can alternate between $\mathtt{t}$ and $\mathtt{f}$. Our finite DAG restriction excludes this feedback, while fixed processing orders determine event creation.

A.3 Position Relative to LLM Evaluation

Existing benchmarks specify different aspects of reasoning correctness. They evaluate proof construction and inference patterns (Tafjord et al., 2021; Parmar et al., 2024), conclusions after successive knowledge changes (Xiu et al., 2022; Helwe et al., 2025), and acceptance under conflicting rules or multiple extensions (Kazemi et al., 2023; Xiu & Liu, 2025). These settings make explicit how available information and inference policies determine warranted conclusions.

Complementary evaluations examine how new evidence affects the strength or acceptance of earlier inferences (Rudinger et al., 2020; Allaway & McKeown, 2025; Wilie et al., 2024), or whether conversational memory reflects updated information (Wu et al., 2025; Chao et al., 2026). Their requirements differ according to how support is supplied or interpreted and whether the evaluation concerns inference strength, accepted beliefs, or subsequent behavior. Table 2 summarizes these distinctions.

Consistency across related beliefs is evaluated by MQuAKE and RippleEdits (Zhong et al., 2023; Cohen et al., 2024), while Grounded Continuation tracks dependencies to propagate retractions and verify responses (He et al., 2026). Our comparison fixes the counterfactual and support cut, separating the choice of support to remove from how the remaining state is maintained. We examine both current values and surviving application histories that can affect later withdrawals. Relational facts can also be represented as propositions; the distinction concerns how support is retained or reconstructed and how it governs subsequent outcomes.

Evaluation must therefore specify the maintenance policy before judging model responses. A symbolic engine computes reference answers under that policy, including the effects of support withdrawal, preservation of alternatives, and downstream reassessment. We evaluate each procedure against its own reference answers and analyze agreement between procedures separately. This distinguishes differences permitted by the maintenance semantics from failures to carry out the designated procedure.

Table 2: Semantic targets of selected benchmarks and maintenance systems. Structures are schematic; $\Rightarrow$ denotes defeasible inference.

Setting	**Reasoning structure**	**Semantic target**
Defeasible NLI	Context, hypothesis, new evidence	Strengthening or weakening of support.
Belief-R	Natural-language conditionals	Additional requirements versus alternative sufficient conditions.
DEFREASING	$S \subseteq K$, $K \Rightarrow A$	Strength of property inheritance $S \Rightarrow A$.
STALE	Memories and later observations	Implicit obsolescence and subsequent behavior.
LogicNMR	Default theory (W, D)	Conclusions after successive fact additions.
ReviseQA	Facts and rules across turns	Conclusions after additions and removals.
BoardgameQA	Conflicting rules and preferences	Acceptance under supplied priorities.
MultiLogicNMR	Extensions $\mathcal{E}(K)$	Skeptical or credulous acceptance.
MQuAKE / RippleEdits	Relational facts and edits	Consistent updates to related conclusions.
Grounded Continuation	Claim dependencies and validity	Retraction propagation and response verification.
This work	Histories $H_t(p)$ and a shared cut C	Current and subsequent outcomes under specified maintenance procedures.

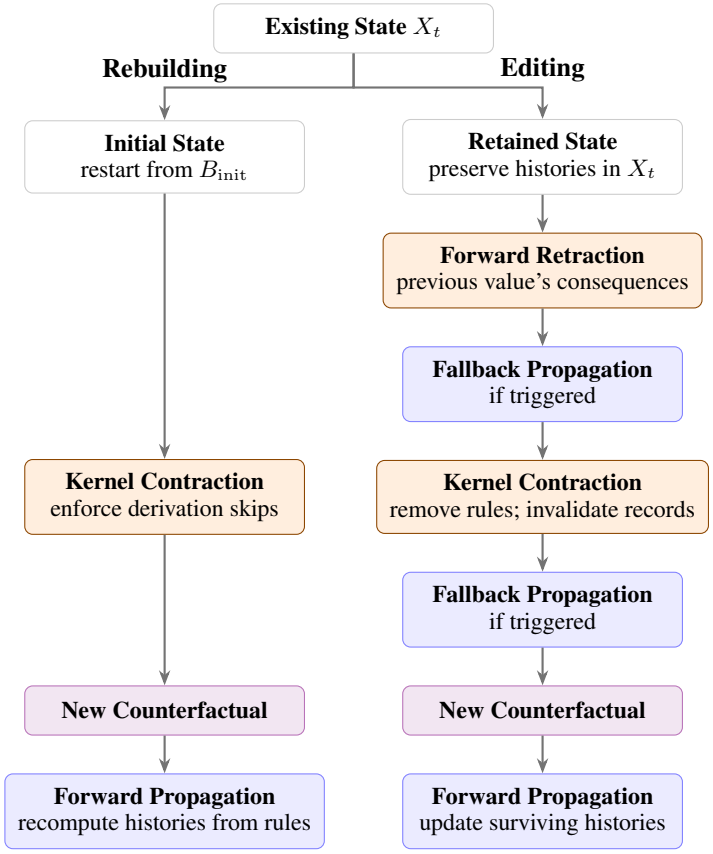


Figure 8: Rebuilding and editing from the same existing state X_t. Rebuilding restarts from the initial belief base and recomputes derived histories under the selected cuts and counterfactual. Editing maintains the existing histories through retraction and propagation. Orange denotes retraction or contraction, purple counterfactual insertion, and blue propagation.

B Theoretical Analysis and Proofs

B.1 Kernel Contraction and Processing Order

Retraction at an intermediate proposition q in the DAG affects both its upstream support and downstream consequences. Let ℓ_{old} denote its previously resolved literal. Following Figure 8, we describe forward retraction, kernel selection and incision, and propagation.

Forward retraction. Editing first invalidates existing applications that depend on ℓ_{old} and follows the resulting loss of support along outgoing dependencies. Each affected proposition is resolved from its last surviving valid application, or becomes u if none remains. Value changes exposed by historical fallback propagate through further invalidation and fallback only; newly enabled applications are deferred to inference. During retraction, q is temporarily excluded as a premise source, while its history is retained. Rebuilding skips this stage because it constructs fresh histories rather than maintaining existing applications.

Example. Suppose $q = \mathsf{t}$ and $r : q \to p$ is the only support for p. Withdrawing the previous value of q in preparation for the counterfactual $q = \mathsf{f}$ invalidates the application deriving p:

$$H_{\text{before}}(p) = \langle (r, 1) \rangle, \quad \text{resolve}_{\text{before}}(p) = \mathsf{t},$$
$$H_{\text{after}}(p) = \langle (r, 0) \rangle, \quad \text{resolve}_{\text{after}}(p) = \mathsf{u}.$$

Thus, withdrawing q's previous value also retracts p. The record remains in the history with validity 0, while the rule itself remains available.

Kernel selection. We adapt the incision principle of kernel contraction (Hansson, 1994) to the current resolved state. A kernel for ℓ_{old} is an inclusion-minimal set of available rules deriving it from empty-premise rules, with every derived literal matching X_t. The kernel family $\mathcal{K}_t(\ell_{\text{old}})$ is fixed before retraction, excluding support enabled only by later fallback. A support cut C is selected from the available rules and satisfies

$$C \cap K \neq \varnothing \qquad \text{for every } K \in \mathcal{K}_t(\ell_{\text{old}}). \tag{16}$$

The cumulative cut may additionally remove initially inapplicable rules deriving the selected initial literals. Thus, C interrupts all current support but need not be contained in the union of current kernels. It does not preclude restoration through fallback. Rebuilding excludes C during reconstruction; editing also invalidates its existing applications.

Example. Rules $r_0 : \varnothing \to a$ and $r_1 : a \to q$ support $q = \mathsf{t}$. For a counterfactual $q = \mathsf{f}$:

$$\begin{aligned} \text{Kernel selection:} \quad & K = \{r_0, r_1\}, \\ \text{Incision:} \quad & C = \{r_1\}, \qquad (r_1, 1) \longrightarrow (r_1, 0). \end{aligned}$$

The incision removes r_1 from availability. In editing, $(r_1, 0)$ is its retained but invalid application record; cutting r_0 instead withdraws support further upstream.

Propagation. Forward retraction or support contraction may expose an earlier valid application with a different resolved value. During either retraction phase, the change propagates along outgoing dependencies by invalidating incompatible applications and resolving further fallback. No new application is created during these phases. Surviving records retain their positions, and newly enabled applications are deferred to subsequent inference. Upstream cuts may also affect branches outside the target's descendants.

Example. Let $r_0 : \emptyset \to \neg p$, $r_1 : q \to p$, and $r_2 : \neg p \to s$. Initially, $H_t(p) = \langle (r_0, 1), (r_1, 1) \rangle$ and $H_t(s) = \langle\rangle$. In preparation for the counterfactual $q = \mathsf{f}$, forward retraction invalidates the application of r_1 and exposes r_0:

$$\begin{aligned} H_{\text{ret}}(p) &= \langle (r_0, 1), (r_1, 0) \rangle, & \text{resolve}_{\text{ret}}(p) &= \mathsf{f}, \\ H_{\text{ret}}(s) &= \langle\rangle, & \text{resolve}_{\text{ret}}(s) &= \mathsf{u}. \end{aligned}$$

The premise of r_2 is now satisfied, but its application is deferred to inference. If the cut preserves r_0 and r_2, inference after counterfactual insertion appends the new application:

$$H_{\text{inf}}(s) = \langle (r_2, 1) \rangle, \qquad \text{resolve}_{\text{inf}}(s) = \mathsf{t}.$$

After both retraction phases, editing inserts the counterfactual, restores q as a premise source, and performs inference over the surviving histories. Ordinary inference first propagates fallback values and newly enabled support from affected upstream propositions. Counterfactual-grounded support is then propagated in a final pass under the shared priority and fixed processing order. Rebuilding constructs fresh histories from the revised initial state under the same cut and counterfactual.

Counterfactual paths. *On-CF-Path* comprises the counterfactual target and propositions inferred from its imposed value through successive applications of surviving rules. All other propositions are *Off-CF-Path*. For example, under $p \to q$, $q \to r$, and $r \to s$, imposing $p = \mathsf{t}$ derives q, r, and s, placing them On-CF-Path.

Semantic considerations.

- **Order dependence.** Backward and forward retraction need not commute: their order can change fallback and propagation. Applying the counterfactual last under the shared priority and propagation order yields agreement along the same CF-induced derivations, while other values may differ. We therefore evaluate On-CF-Path and Off-CF-Path separately.
- **Withdrawal and restoration.** Cutting current support may expose an alternative path, restoring a withdrawn value even in a DAG. For example, let $r_0 : \varnothing \to \neg a$, $r_1 : \varnothing \to a$, $r_2 : a \to q$, and $r_3 : \neg a \to q$. With $H_t(a) = \langle (r_0, 1), (r_1, 1) \rangle$, the current kernel for q is $\{r_1, r_2\}$. Cutting r_1 exposes r_0 and enables r_3, restoring support for q. If $q = \mathsf{f}$ is imposed first, counterfactual priority preserves that value but does not prevent regeneration of its opposing support.

B.2 Proofs of Theoretical Results

We use the history, inference, and counterfactual-priority semantics defined in the main text. Editing performs forward retraction, backward retraction, counterfactual insertion, and inference in that order (Equation 9). Both retraction phases preserve history positions and only invalidate existing applications. Newly supported applications are appended during inference, while continuously supported applications retain their records unless renewed as CF-grounded support. All comparisons below concern closed states under the same fixed topological and rule order.

Proposition 1 (Multiplicity of support retraction). Consider an initially supported, isolated chain

$$\ell_0 = \alpha \xrightarrow{r_1} \ell_1 \xrightarrow{r_2} \cdots \xrightarrow{r_k} \ell_k = \beta, \qquad k \geq 2. \tag{17}$$

The initial support $r_0 : \varnothing \to \ell_0$ is fixed. Assume that the chain propositions have no other incoming support, including support available through historical fallback.

A rule retraction removes only the selected rule. An intermediate-literal retraction recursively removes its upstream support rules, stopping at the fixed initial support. Their respective cuts are

$$\begin{aligned} C_i^{\mathrm{rule}} &= \{r_i\}, & 1 \leq i \leq k, \\ C_i^{\mathrm{lit}} &= \{r_1, \ldots, r_i\}, & 1 \leq i < k. \end{aligned} \tag{18}$$

Both operations include downstream invalidation and subsequent maintenance to closure. Let

$$\mathcal{C}_\pi = \{C_i^{\mathrm{rule}} \mid 1 \leq i \leq k\} \cup \{C_i^{\mathrm{lit}} \mid 1 \leq i < k\},$$

and let X_C denote the state obtained after applying C to the same initial state, before counterfactual insertion. We group cuts by the current values of their resulting states:

$$C \sim_{\mathrm{val}} D \iff X_C \equiv_{\mathrm{val}} X_D.$$

The family $\mathcal{C}_\pi$ contains $2k-2$ distinct rule cuts, all of which retract β. These cuts form k current-value equivalence classes. One class contains $k-1$ cuts, while each of the remaining $k-1$ classes contains a single cut.

Proof. There are k singleton rule cuts and $k-1$ supporting-prefix cuts. Their only overlap is

$$C_1^{\mathrm{rule}} = C_1^{\mathrm{lit}} = \{r_1\}.$$

Every other literal cut contains at least two rules, and the literal cuts have different cardinalities. Consequently,

$$|\mathcal{C}_\pi| = k + (k-1) - 1 = 2k - 2.$$

For illustration, when $k = 3$, the four distinct cuts are shown below. A cross marks a removed rule; an unmarked rule may remain available while its application becomes invalid.

$$\begin{aligned} \{r_3\}: &\quad \ell_0 \xrightarrow{r_1} \ell_1 \xrightarrow{r_2} \ell_2 \xrightarrow[\times]{r_3} \ell_3, \\ \{r_2\}: &\quad \ell_0 \xrightarrow{r_1} \ell_1 \xrightarrow[\times]{r_2} \ell_2 \xrightarrow{r_3} \ell_3, \\ \{r_1\}: &\quad \ell_0 \xrightarrow[\times]{r_1} \ell_1 \xrightarrow{r_2} \ell_2 \xrightarrow{r_3} \ell_3, \\ \{r_1, r_2\}: &\quad \ell_0 \xrightarrow[\times]{r_1} \ell_1 \xrightarrow[\times]{r_2} \ell_2 \xrightarrow{r_3} \ell_3. \end{aligned}$$

The first two cuts preserve different prefixes. The last two both leave only ℓ_0 supported, despite removing different rule sets.

For the general case, define the earliest removed rule by

$$j(C) = \min\{i \mid r_i \in C\}.$$

The rules supporting $\ell_0, \ldots, \ell_{j(C)-1}$ remain available, so this prefix retains its initial values. Removing $r_{j(C)}$ withdraws support for $\ell_{j(C)}$, and invalidation propagates through the suffix to β. With no alternative support, every proposition in this suffix resolves to u.

Writing $p_i = \mathrm{atom}(\ell_i)$ and X for the initial state, we therefore have

$$\mathrm{resolve}_{X_C}(p_i) = \begin{cases} \mathrm{resolve}_X(p_i), & 0 \le i < j(C), \\ \mathsf{u}, & j(C) \le i \le k. \end{cases}$$

Hence two cuts yield the same current values exactly when their earliest removed rules coincide:

$$C \sim_{\mathrm{val}} D \iff j(C) = j(D).$$

Every literal cut contains r_1 and is therefore current-value equivalent to C_1^{rule}. This gives the class

$$\mathcal{E}_1 = \{C_1^{\mathrm{rule}}\} \cup \{C_i^{\mathrm{lit}} \mid 2 \le i < k\}, \qquad |\mathcal{E}_1| = k - 1.$$

For each $j = 2, \dots, k$, the only cut in $\mathcal{C}_\pi$ whose earliest removed rule is r_j is C_j^{rule}. Thus the remaining classes are

$$\mathcal{E}_j = \{C_j^{\mathrm{rule}}\}, \qquad 2 \le j \le k.$$

There are therefore k equivalence classes: one containing $k-1$ cuts and $k-1$ singleton classes. □

Scope of the count. The $2k-2$ cuts are alternative choices applied to the same initial state, not successive cumulative levels. Cumulative application combines removed-rule sets by union and may introduce no additional change when a later cut is already included. Moreover, current-value equivalence does not imply identical surviving rule sets or identical responses to future updates.

Proposition 2 (Regional value equivalence). Let

$$E = \mathrm{Edit}_{\Delta,C}(X_t), \qquad R = \mathrm{Rebuild}_{\Delta,C}(X_t),$$

where X_t is constructed from the initial belief base under the fixed inference order. The procedures use the same cut C, counterfactual Δ, surviving rules, and inference order, and terminate in closed states. We distinguish the following regions:

1. *Upstream region $\mathcal{U}_C$:* propositions preceding both the target q and every cut location in the topological order. A cut location is the head proposition of a rule in C.
2. *Contraction-only region $\mathcal{B}_C$:* propositions whose edited histories change only through invalidation of cut-rule applications. Uncut incoming rules retain their original applicability in both procedures, without additional renewal caused by intermediate value changes.
3. *On-CF-Path region $\mathcal{F}_\Delta$:* the target and propositions supported at closure by valid applications of the common counterfactual propagation. These applications have final priority over competing ordinary applications in both procedures.

Under these conditions,

$$\mathrm{resolve}_E(p) = \mathrm{resolve}_R(p) \qquad \forall p \in \mathcal{U}_C \cup \mathcal{B}_C \cup \mathcal{F}_\Delta.$$

Proof. We establish equality separately for the three regions.

Upstream region. A proposition $p \in \mathcal{U}_C$ precedes the target and every cut location. Neither the counterfactual nor forward propagation from a cut location can therefore reach p. The same holds for its predecessors in the DAG. Its incoming rules and their supporting premises are unaffected by the revision.

Induction over the topological order shows that rebuilding reproduces the original applications in this region. Editing preserves those applications and their order. Both procedures consequently reproduce the values in X_t.

Contraction-only region. For $p \in \mathcal{B}_C$, editing changes the history only by invalidating applications of rules in C. Its valid rule sequence is therefore

$$H_E^{\mathrm{valid}}(p) = H_t^{\mathrm{valid}}(p)\big|_{\mathcal{R}\setminus C},$$

where the restriction preserves the order of the remaining applications.

Rebuilding excludes the same rules. By the applicability condition, each uncut incoming rule contributes an application exactly when it did in the original construction, without additional renewal from intermediate changes. Since X_t and R use the same fixed inference order, rebuilding reproduces the same remaining rule sequence:

$$H_R^{\mathrm{valid}}(p) = H_E^{\mathrm{valid}}(p).$$

The application records themselves may have different creation times, but their relative order and conclusions agree. Both procedures therefore select the same governing application. If the sequence is empty, both assign u.

On-CF-Path region. At q, the common protected counterfactual determines the same value in both states. Proceed in topological order along the common counterfactual propagation. Suppose the preceding CF-supported propositions have the same resolved values. Because rules have a single premise, the same surviving rules are supported by these CF-derived literals in both procedures. The shared propagation order consequently produces the same ordered CF-induced applications.

For a reached proposition in $\mathcal{F}_\Delta$, the common sequence contains valid applications at closure. By the priority assumption, competing ordinary applications cannot override this sequence. Its last valid application therefore determines the same resolved value in both states. Induction establishes equality throughout $\mathcal{F}_\Delta$.

Together, the three cases establish regional value equivalence. Outside their union, invalidation, fallback, and renewed applicability may produce different application orders, so the result does not guarantee equality over the entire belief state. □

Theorem 1 (Value equivalence is not sequential equivalence). For a subsequent update $\Gamma = (\Delta_\Gamma, C_\Gamma)$, write

$$\mathrm{Ret}_\Gamma = \mathrm{Edit}_{\Delta_\Gamma, C_\Gamma}.$$

Thus, the subsequent retraction maintains existing histories under the specified editing semantics. *There exist historical states X, Y and a common subsequent retraction Γ such that*

$$X \equiv_{\mathrm{val}} Y, \qquad \mathrm{Ret}_\Gamma(X) \not\equiv_{\mathrm{val}} \mathrm{Ret}_\Gamma(Y).$$

Proof. Consider the propositions $\{a, b, c, p\}$ and the rules

$$s_a : \varnothing \to a, \quad s_b : \varnothing \to b, \quad s_c : \varnothing \to c,$$
$$r_+ : a \to p, \quad r_- : b \to \neg p, \quad r_c : c \to p.$$

Both states use this rule set and have identical source histories, each containing its corresponding valid empty-premise application. Thus, $a = b = c = \mathsf{t}$ in both states.

Let the histories at p be

$$H_X(p) = \langle (r_+, 1), (r_-, 1), (r_c, 1) \rangle,$$
$$H_Y(p) = \langle (r_-, 1), (r_+, 1), (r_c, 1) \rangle.$$

All applications at p are ordinary and have the same priority. The different orders represent different earlier support episodes: a and b became supported in opposite orders, followed by c. All three applications are currently valid, and the common latest application r_c governs both histories. Hence $p = \mathsf{t}$ in both states, giving $X \equiv_{\mathrm{val}} Y$.

Let Γ use the common cut $C_\Gamma = \{s_c\}$ and impose the counterfactual $c = \mathsf{f}$. The old support for $c = \mathsf{t}$ is withdrawn, and the application of r_c becomes invalid. The premises of r_+ and r_- remain satisfied throughout the update. These applications therefore retain their original positions:

$$H_{\mathrm{Ret}_\Gamma(X)}(p) = \langle (r_+, 1), (r_-, 1), (r_c, 0) \rangle,$$
$$H_{\mathrm{Ret}_\Gamma(Y)}(p) = \langle (r_-, 1), (r_+, 1), (r_c, 0) \rangle.$$

No rule has premise $\neg c$, so the new counterfactual enables no application at p. Moreover, inference does not append new copies of the continuously supported applications of r_+ and r_-. The displayed histories therefore remain unchanged at closure.

The first state falls back to r_- and resolves p to f. The second falls back to r_+ and resolves p to t. Consequently,

$$\mathrm{Ret}_\Gamma(X) \not\equiv_{\mathrm{val}} \mathrm{Ret}_\Gamma(Y).$$

The difference follows from the order of surviving valid applications, despite agreement on all values before the common update. □

Remarks.

- **Correctness depends on the procedure.** A counterfactual fixes the target's new value but does not uniquely determine the remaining belief state. Support selection, retained histories, inference order, and counterfactual priority also determine the outcome. Model predictions should therefore be evaluated under the procedure specified in their instructions. Scoring one procedure against another's references on disagreement cases measures reference agreement, rather than correctness under the instructed procedure (Appendix D.4).
- **Shared cuts and retained histories.** Editing and rebuilding use the same rule cut C, selected from the pre-retraction state. The path-hitting condition in Equation 7 concerns current support paths; later fallback may expose or enable other support. Fixing C separates support selection from how the remaining state is maintained. The relevant historical information is which valid applications survive and in what order. Retaining invalid records alone does not establish sequential divergence. The theorem concerns historical states with different prior episodes; it does not require them to arise from editing and rebuilding a common initial state.
- **Computational cost.** Editing can exploit an existing state when only a small region is affected. However, support analysis, invalidation, fallback, and subsequent inference may revisit overlapping dependencies. Repeated updates can also increase the number of retained application records. The computational cost therefore depends on the affected region, history representation, and implementation. The semantic results establish no general runtime advantage for either procedure.

C NMR-RETRACTION DATASET

C.1 DATA GENERATION

NMR-Retraction evaluates how language models maintain inferred beliefs after a counterfactual intervention. Each instance consists of a signed rule graph, a counterfactual, a specified support cut, and a query proposition. A deterministic symbolic engine computes support histories and procedure-specific answers, enabling matched comparisons from the same initial state. The dataset includes independent backward and forward retraction tasks and comparisons between Rebuild, Backward Edit, and Edit.

Graph and intervention generation.

- **Graph structure.** We generate six-layer directed acyclic graphs (L_0–L_5) with 18–30 propositions. As illustrated in Figure 9, L_0 contains source propositions, L_1–L_2 provide upstream support for an intervention at L_3, and L_4–L_5 capture its downstream effects.
- **Rules and initial values.** Signed unary rules connect adjacent layers, with 2–4 incoming rules per derived proposition. Source values and rule-sign combinations are approximately balanced; conflicts, fallback supports, and resolved values are determined by symbolic inference.
- **Counterfactual intervention.** We select a target at L_3 whose initial value is T or F and insert a protected counterfactual asserting the opposite value.
- **Evaluation.** Cumulative cuts extend backward along all initial support paths for the target's previous value, while preserving opposite-polarity supports. This construction allows us to evaluate both the removal of upstream support and the subsequent maintenance of downstream beliefs. Each cut configuration and computation method is evaluated independently from the same initial state. Within each pool, no graph structure is shared between the training and test splits.

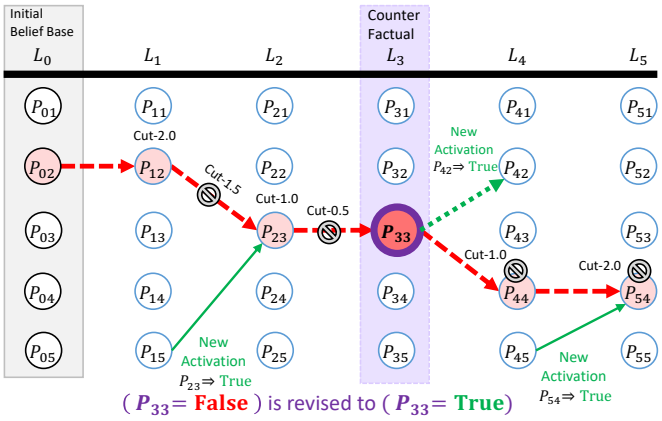


Figure 9: Six-layer graph used to construct retraction instances. A protected counterfactual reverses the initial value of a target at L_3. Upstream paths illustrate support for its previous value and candidate cut locations; downstream paths illustrate dependencies through which retraction and renewed inference can affect beliefs at L_4–L_5. Only selected rules are shown for clarity.

Instance selection and annotation. The directional pool isolates backward and forward invalidation and fallback without new inference, under locality and no-re-entry constraints. The propagation pool requires at least one cut with maintenance effects on other nodes, a CF-governed proposition, and an affected proposition without valid CF-grounded support. Candidates are not selected based on disagreement between methods. The engine records procedure-specific final values; Edit defines the propagation regions and BM/BU categories. For each propagation split and region, we select at most one BM and one BU query per graph, balancing available layer–cut cells within each category. Algorithm 1 summarizes construction.

Algorithm 1 NMR-Retraction dataset construction

Require: Pool type P, split sizes, width–density configurations
Ensure: Matched retraction queries with procedure-specific answers
Generation goals
Directional: isolate backward and forward retraction.
Propagation: include CF-governed beliefs and affected beliefs without valid CF-grounded support.
Generation
for each graph slot **do**
  Sample a signed layered graph and compute initial values and support histories
  Select a target at L_3, reverse its value, and construct cuts from all initial support paths
  Resample until the pool's acceptance criteria hold
  Compute primitive or Rebuild/Edit endpoints independently from the same initial state
  Store procedure-specific answers and query categories
end for
Retain cuts $\{0.5, 1.0, 2.0\}$ and independent Forward queries
For each propagation split and region, select at most one BM and one BU query per graph, balancing layer–cut cells
Verification
Check graph uniqueness within each pool across splits
Verify intervention consistency and symbolic endpoint replay
Check answer agreement in the shared CF-governed region

C.2 Data Statistics

Evaluation sets and retraction. We use 900 graphs (2,907 queries) for parameter tuning (**P-Train**) and 1,800 graphs (5,755 queries) for procedure and prompting evaluations (**P-Test**). For a separate analysis of controlled cuts along retraction paths, we construct **R-Test** with 360 graphs. Each R-Test graph has six layers (L_0–L_5), with 3, 4, or 5 propositions per layer and low, medium, or high rule complexity; each of the nine width–complexity combinations contains 40 graphs. Backward retraction applies cumulative cuts at depths $\{0.5, 1.0, 2.0\}$ to 360, 270, and 90 eligible graphs, respectively, always starting from the same initial state. Forward retraction is evaluated independently on all 360 graphs without a backward cut. R-Test contains 4,115 queries, including BM, BU, and counterfactual-target queries; its labels follow the corresponding retraction primitive (Table 3).

Table 3: Query counts by evaluation set and task. CF denotes counterfactual-target queries. Total rows sum the retraction tasks or propagation regions, respectively. Counts exclude repetitions across models, prompts, and procedures.

Set	Task / region	BM	BU	CF	Total
R-Test	Backward retraction	1,267	466	720	2,453
	Forward retraction	624	678	360	1,662
	Retraction total	1,891	1,144	1,080	4,115
P-Test	CF-grounded	1,105	1,499	0	2,604
	Beyond CF propagation	1,351	1,800	0	3,151
	Propagation total	2,456	3,299	0	5,755
P-Train	CF-grounded	561	751	0	1,312
	Beyond CF propagation	695	900	0	1,595
	Propagation total	1,256	1,651	0	2,907

Rebuild vs. Edit. On P-Test, we compare Rebuild, Backward Edit, and Edit. Edit defines two evaluation regions: propositions governed by CF-grounded support, excluding the counterfactual target, and propositions whose values or histories change without valid CF-grounded support. The latter excludes unaffected and mixed-support propositions. Within each region, we select at most one BM and one BU query per graph across the three evaluated cuts, balancing available layer–cut cells separately for each category. The 5,755 queries cover 5,638 unique propositions. BM/BU membership is fixed across procedures: BM preserves the initial value despite a change in support history, while BU includes all value changes, including transitions to or from U. Primary accuracy uses each procedure's own gold answer; agreement with Edit is measured separately. The same queries support the belief-oriented prompting evaluation.

C.3 LLM PROMPTS

Task principles and computation methods. The task prompt defines shared principles of belief maintenance, including support validity, historical fallback, counterfactual priority, and deterministic resolution. The editing prompt specifies a procedure under these principles, whose scope, history handling, and stopping condition determine the resulting state. Edit defines our reference labels. Rebuild and Backward Edit agree with this reference where the same counterfactual-grounded support governs, but may differ elsewhere. Thus, shared principles and identical cuts do not imply identical outcomes.

Property	Rebuild	Backward Edit	Edit
Shared task principles	✓	✓	✓
Retains derived histories	–	✓	✓
Explicit upstream maintenance	–	✓	✓
Counterfactual-grounded propagation	✓	✓	✓
Explicit downstream invalidation	–	–	✓

Prediction strategies and composition. Direct requests only the final value; CoT adds a brief support-based explanation. Variants within the Editing and Prediction boxes are grouped for presentation but used separately: each condition combines the shared task prompt with one computation method and one prediction strategy.

Editing Prompt

1. Computation method: Rebuild

- Form the revised input by adding the protected counterfactual and applying the supplied cumulative cut to the initial facts and rules. Preserve all facts and rules that remain available.
- Discard previously derived values and rule-application histories. Do not reuse an earlier inference as a source fact or retain its support merely because it existed before the intervention.
- Start a fresh computation from the surviving source facts and the protected counterfactual.
- Reconstruct derived values and support histories by processing layers in increasing order and incoming rules in their supplied order. Determine rule applicability from the values resolved during this new computation.
- Resolve conflicting supports using the specified priorities and birth order. An earlier conclusion survives only if the revised input derives it again; otherwise, use the newly governing value or `U` when no valid support exists.
- Complete inference over the entire graph to closure, then report the query's resolved value.

2. Computation method: Backward Edit

- Start from the initial belief state, retaining its resolved values and ordered support histories.
- Insert the protected counterfactual, then apply the supplied cumulative cut by disabling exactly the listed facts and rules simultaneously.
- Maintain the affected upstream state in increasing layer order. Invalidate unsupported applications, resolve fallback from surviving supports, and perform newly enabled upstream inference.
- Propagate CF-grounded support from the counterfactual target through applicable rules. Create or renew application episodes according to the specified history rules.
- Stop after upstream maintenance and CF-grounded propagation. Do not perform a separate downstream invalidation/fallback sweep or complete ordinary downstream inference.
- Preserve values and histories not modified by these operations, and report the query's resolved value in the resulting state.

3. Computation method: Edit

- Start from the initial belief state, retaining its resolved values and ordered support histories.
- Temporarily exclude the counterfactual target as a premise source. Perform downstream invalidation and fallback in increasing layer order. Invalidate existing applications whose justification no longer holds, including those affected by the target's previous value. Preserve surviving alternative supports. Do not create applications or renew episodes during this invalidation-only stage.
- Apply the supplied cumulative cut by disabling exactly the listed facts and rules simultaneously.
- Maintain the affected upstream state in increasing layer order. Invalidate unsupported applications and resolve fallback from surviving supports. Propagate resulting value changes along outgoing dependencies through further invalidation and fallback. Do not create applications or renew episodes during this stage.
- Insert the protected counterfactual and restore its target as a premise source.
- Perform ordinary and CF-grounded inference to closure. First perform ordinary inference, including newly enabled inferences arising from fallback values and other changes in resolved values. Then propagate CF-grounded support from the counterfactual target through applicable rules in a final pass. Create or renew application episodes according to the specified history rules.
- Preserve the ordered histories throughout these updates, and report the query's resolved value in the resulting state.

Task Instructions

Determine the queried proposition's value after the specified intervention, following the assigned computation method and the definitions below.

1. Beliefs and rules

- Each proposition has a resolved value of `T` (true), `F` (false), or `U` (undetermined).
- An available rule applies when its signed body matches the premise's resolved value: a positive body requires `T`, and a negative body requires `F`. `U` satisfies neither.
- Only a proposition's resolved value can satisfy downstream rule bodies.

2. Rule histories and retraction

- Rule applications are stored as support records with a birth order. A proposition's history may contain multiple records, including conflicting and invalid supports.
- The highest-priority valid support governs the proposition's value. Among supports of equal priority, the newest birth order wins. If no valid support remains, the value is `U`.
- A losing support may remain valid and become governing when a higher-priority support is withdrawn. A stored record does not justify a value merely because it remains in the history.
- The assigned computation method determines whether existing histories are retained and updated or reconstructed from the revised inputs.

3. Belief maintenance

- During validity maintenance, invalidate an application if its rule is unavailable or its signed premise no longer matches the premise's resolved value.
- Resolve the affected proposition from surviving valid supports, including fallback supports. A support change need not change the resolved value.
- A change in resolved value may invalidate downstream applications or enable new inference. Retract supports that lose their justification while preserving independent valid supports.
- Perform maintenance only within the scope and in the order specified by the assigned method. Invalidation-only stages neither create applications nor renew support episodes.

4. Counterfactual intervention

- A counterfactual introduces a protected value at its target. Insert it at the stage specified by the assigned computation method.
- Apply the cumulative cut by disabling exactly the listed facts and rules simultaneously. Do not introduce additional cuts, facts, or rules.
- Support priority is: protected counterfactual support at its target, then CF-grounded support, then ordinary support.
- CF-grounded support propagates through applicable rules whose premises are governed by the protected counterfactual or CF-grounded support.
- During inference, renew an existing ordinary application as a new CF-grounded episode when its premise is governed by the protected counterfactual or CF-grounded support, even if the premise value is unchanged. Preserve the old episode as superseded history. Do not renew an already valid CF-grounded episode again.

Cut depth: cumulative upstream retraction

- Cut depth is measured upstream along the initial support paths for the target's pre-counterfactual value. It is not the number of rules removed or the depth of subsequent forward propagation.
- Relative to the counterfactual target, the depths mean:
  - **0.5:** Remove the supporting rules that directly derive the target's initial value.
  - **1.0:** Include the 0.5 removal and retract support for the initial literals used as premises by those target-near rules.
  - **1.5:** Include the 0.5 and 1.0 removals and remove the next upstream supporting rules that derive those premise literals along the initial support paths.
  - **2.0:** Include the 0.5, 1.0, and 1.5 removals and retract support for the initial literals used as premises by those next-upstream rules.
- Retracting a literal's support means disabling all initially available facts and rules deriving that proposition with its initial polarity, including rules that were not initially applicable. Preserve opposite-polarity supports. Do not delete the proposition or force its value to `U`; resolve its value from surviving valid supports when the assigned method processes it.
- The cut covers all initial support paths for the target's old value, including parallel supporting rules, not just the newest or a single selected path. The protected counterfactual support is not removed.
- Determine all removal targets from the same initial belief state. The supplied fact and rule IDs give the complete cumulative removal set; disable them simultaneously. Do not recompute the cut from intermediate values or add removals beyond the supplied list.
- Different depths may produce the same removal set: the rules specified at 1.5 may already have been disabled by the literal-support removal at 1.0. Do not invent further removals to force different outcomes.
- Each cut condition starts independently from the same initial state and counterfactual, rather than continuing from another condition's endpoint. If the assigned task specifies no cut, do not apply one.

5. Initial belief state and inference

- The initial belief state contains the proposition values and ordered application histories established from the initial facts and rules before the intervention.
- During inference, process layers in increasing order and incoming rules in their supplied order.
- Apply rules using current resolved values, including values established by the counterfactual or fallback.
- A continuously applicable ordinary application retains its birth order when its premise's governing support switches between ordinary supports without changing the premise value.
- Follow the assigned method's scope and stopping condition, and report the query's value at that endpoint.

Prediction: Direct and CoT

1. Direct

Return only one JSON object with exactly one key, `"value"`. The value must be `"T"`, `"F"`, or `"U"`. Do not include an explanation, a code fence, or any text outside the JSON object.

Output format example:
`{"value":"T"}`

The example value is a placeholder, not the answer to the query.

2. CoT

Return only one JSON object with exactly two keys, `"reason"` and `"value"`. In `"reason"`, briefly identify the decisive invalidated, surviving, or newly created support that determines the query's final value. If no valid support remains, state that briefly. Keep the reason within 500 tokens. The value must be `"T"`, `"F"`, or `"U"`. Do not include a code fence or any text outside the JSON object.

Output format example:
`{"reason":"short decisive support trace","value":"T"}`

The example reason and value are placeholders, not the answer to the query.

Model Input

Input graph

Layers in increasing order:
<ordered_layers_and_proposition_ids>

Source facts: IDs, signed values, and initial availability: <source_facts_with_ids_values_and_availability>

Initial fact order: <initial_fact_order>

Rules in fixed order: IDs, signed bodies, and signed heads: <rules_with_ids_order_signed_bodies_and_heads>

Initial state

Initial values and ordered histories, if provided: <b0_values_and_histories_or_not_provided>

Initially unavailable rule IDs: <initially_unavailable_rule_ids>

Initially unavailable fact IDs: <initially_unavailable_fact_ids>

Intervention

Insert the protected counterfactual rule: <cf_rule_id>: $\varnothing \rightarrow$ <cf_target> = <cf_new_value>.

Cut depth: <cut_depth_or_none>

Rule IDs disabled by this intervention: <cut_rule_ids>

Fact IDs disabled by this intervention: <cut_fact_ids>

Query
<query_proposition_id>

D EXPERIMENTS

D.1 EXPERIMENTAL SETTINGS

Implementation and generation. Evaluation uses vLLM 0.11.0 in BF16 on NVIDIA B200 GPUs. Models up to 32B use one GPU; Llama-3.3-70B uses two-way tensor parallelism. For both Direct and CoT, we use greedy decoding with temperature 0, top-p 1.0, and a maximum of 8,192 generated tokens. We generate one response per query using the model's chat template and generation seed 0. Unless otherwise specified, experiments use *initial-only* inputs, providing facts and rules without initial belief values or support histories. Responses must follow the prescribed JSON format. We score the final value and count malformed responses as incorrect; the logical correctness of generated explanations is not separately scored.

D.2 RETRACTION PREDICTION ACCURACY

Table 4 extends the main-text comparison to all eight models. Backward and forward retraction are evaluated independently from the same initial state, without subsequent inference. The results highlight two patterns:

- **CoT improves retraction on average.** Mean backward BU accuracy increases from 30.9% to 55.5%, and forward BU accuracy from 13.3% to 38.4%.
- **Acceptance does not ensure successful retraction.** Llama-3.3-70B reaches 99.7% counterfactual acceptance under CoT, but only 25.3% backward and 19.2% forward BU accuracy.

Table 4: Counterfactual acceptance and independent retraction accuracy (%).

	Counterfactual		Backward (BU)		Forward (BU)	
Model	D	CoT	D	CoT	D	CoT
Qwen3-4B	**94.0**	93.6	5.2	**79.0**	6.0	**49.0**
Qwen3-14B	89.0	**99.9**	55.6	**79.6**	25.4	**61.2**
Qwen3-32B	**100.0**	99.9	**82.8**	62.4	6.6	**56.2**
Gemma3-4B	41.6	**50.6**	43.6	**66.1**	19.6	**39.5**
Gemma3-12B	**80.5**	67.7	28.8	**64.2**	29.5	**31.7**
Gemma3-27B	**99.2**	79.4	16.3	**61.6**	9.0	**44.8**
Llama3.1-8B	52.6	**75.9**	**6.7**	5.6	**5.5**	**5.5**
Llama3.3-70B	98.7	**99.7**	8.4	**25.3**	4.7	**19.2**
Average	81.9	**83.4**	30.9	**55.5**	13.3	**38.4**

D.3 On CF-Path Evaluation

Table 5 compares Rebuild, Backward Edit, and Edit on counterfactual-governed propositions at L_4 and L_5. All procedures share reference answers. Scores use common strata with at least five queries per layer and stratum, and fixed pooled weights computed separately for BU and BM. No procedure consistently performs best across models, prompting conditions, and metrics. Under CoT, accuracy is commonly lower at the second hop, although this pattern is not universal.

Table 5: Standardized On CF-Path BU/BM accuracy (%) with shared reference answers.

		BU				BM			
		Direct		CoT		Direct		CoT	
Model	Method	1-hop	2-hop	1-hop	2-hop	1-hop	2-hop	1-hop	2-hop
Qwen3-4B	Rebuild	**53.6**	**58.7**	65.3	49.6	**66.1**	**69.0**	66.3	48.5
	Backward Edit	53.3	56.8	69.2	48.7	64.9	64.6	70.5	47.6
	Edit	52.6	58.5	**71.6**	**50.4**	65.4	67.1	**70.7**	**50.9**
Qwen3-14B	Rebuild	50.8	55.4	76.5	**55.2**	62.9	66.2	**79.0**	**64.9**
	Backward Edit	46.2	53.6	70.5	47.5	60.0	64.4	74.4	53.3
	Edit	**56.4**	**60.6**	**76.9**	53.8	**67.8**	**72.1**	76.7	58.6
Qwen3-32B	Rebuild	50.5	**52.3**	**80.5**	57.3	61.0	66.8	80.1	66.4
	Backward Edit	46.8	43.7	78.5	53.9	59.6	62.4	**84.1**	64.7
	Edit	**52.1**	49.0	74.4	**58.8**	**66.8**	**69.3**	78.1	**67.5**
Gemma3-4B	Rebuild	47.5	47.4	**38.3**	**38.6**	50.1	49.4	**40.1**	**41.4**
	Backward Edit	**53.5**	**52.2**	31.7	30.4	**58.3**	**54.7**	33.1	28.9
	Edit	49.4	49.5	34.0	32.6	50.8	52.4	38.4	35.7
Gemma3-12B	Rebuild	41.8	41.9	**59.4**	**49.9**	54.7	52.7	**64.0**	**57.6**
	Backward Edit	52.6	56.3	55.2	45.8	68.7	**69.2**	56.9	53.9
	Edit	**54.8**	**57.8**	55.6	48.1	**69.4**	68.2	59.2	54.6
Gemma3-27B	Rebuild	62.9	67.0	74.1	**66.7**	87.1	82.3	84.0	**75.4**
	Backward Edit	**63.0**	65.6	77.6	60.5	**87.7**	82.5	80.0	66.4
	Edit	62.2	**67.0**	**82.8**	65.7	86.2	**84.3**	**85.2**	71.7
Llama3.1-8B	Rebuild	**49.9**	53.5	53.3	56.0	55.1	60.8	54.7	58.7
	Backward Edit	49.0	**55.7**	50.9	**57.9**	**55.7**	**62.2**	**57.9**	**65.9**
	Edit	49.5	52.0	**54.4**	57.0	53.1	58.6	57.8	62.5
Llama3.3-70B	Rebuild	67.5	64.3	72.2	64.7	**88.4**	**81.3**	82.5	**80.5**
	Backward Edit	68.0	64.0	**76.8**	**69.0**	87.7	80.0	**85.7**	79.3
	Edit	**68.8**	**65.3**	75.4	66.2	87.6	80.8	84.9	78.1

Table 6: Edit reference agreement (%) on 77 Off-CF BU queries where EDIT differs from both REBUILD and BACKWARD EDIT, a subset of the 82 EDIT–REBUILD disagreement queries.

	Direct			CoT		
Model	Edit (↑)	Rebuild (↓)	Back. Edit (↓)	Edit (↑)	Rebuild (↓)	Back. Edit (↓)
Qwen3-4B	39.0	39.0	**37.7**	31.2	**28.6**	39.0
Qwen3-14B	37.7	37.7	**36.4**	55.8	**31.2**	45.5
Qwen3-32B	35.1	**31.2**	39.0	39.0	39.0	**29.9**
Gemma3-4B	35.1	**36.4**	39.0	35.1	36.4	**35.1**
Gemma3-12B	29.9	**24.7**	27.3	33.8	**27.3**	**27.3**
Gemma3-27B	36.4	**36.4**	40.3	31.2	42.9	**37.7**
Llama3.1-8B	33.8	**35.1**	36.4	41.6	**28.6**	41.6
Llama3.3-70B	36.4	**33.8**	**33.8**	40.3	**35.1**	37.7

D.4 OFF CF-PATH EVALUATION

Table 6 reports agreement with EDIT on 77 Off-CF BU queries where the EDIT reference differs from both REBUILD and BACKWARD EDIT. These scores measure agreement with a common reference, rather than correctness under each assigned procedure. For EDIT, accuracy ranges from 29.9% to 55.8% across models and prompting conditions, indicating difficulty in revising these selected beliefs outside counterfactual propagation.

Rationale and scope of the off-CF evaluation. Theorem 1 establishes that states with identical current values but different support histories can respond differently to a common subsequent retraction. Evaluating whether models reproduce this behavior requires controlling both the initial value agreement and the histories available for the subsequent update. Our single-update experiment addresses a related question: whether models correctly predict revisions obtained by reassessing existing histories outside counterfactual propagation.

In this synthetic setting, reference labels are generated by symbolically executing a specified revision procedure, while LLM prompts instruct models how to perform the revision. EDIT reassesses retained applications, whereas REBUILD reconstructs histories from the revised belief base. When these procedures prescribe different values, evaluating REBUILD prompts against EDIT labels introduces a mismatch between the procedure used to generate the reference and the procedure the model is instructed to follow. A model can therefore follow its prompt correctly yet disagree with the reference. Cross-scoring measures reference agreement rather than procedure-specific correctness and cannot directly test the history dependence characterized by Theorem 1. We adopt EDIT references because our target capability is reassessment of the existing support history.

For Figure 5, we select 82 BU queries without valid counterfactual-grounded support in the revised EDIT state, whose reference values differ from both their initial values and the REBUILD references. These values are not directly governed by counterfactual priority and must be resolved from the remaining ordinary support. The selection ensures that neither retaining the initial value nor correctly reproducing the REBUILD result satisfies the evaluation target. It thus focuses on changes prescribed by reassessing retained histories beyond counterfactual-grounded inference. Accordingly, EDIT scores measure accuracy under its own semantics, while REBUILD scores provide an agreement comparison against the same references. This comparison does not establish procedural superiority or identify the model's internal reasoning process.

Off-path histories may also affect later counterfactual judgments when subsequent updates expose alternative applications along connecting dependencies, although these later effects are not measured here. Evaluating such behavior across sequential retractions adds the burden of tracking evolving histories and repeatedly performing invalidation, fallback, and propagation. A controlled evaluation must also distinguish errors inherited from earlier updates from failures in subsequent maintenance. Given the difficulty already observed in predicting revised values after a single update, we focus here on single-update maintenance and leave controlled multi-retraction evaluation to future work.

D.5 Supervised Fine-Tuning

We train on 2,907 queries from 900 graphs using Edit labels. SFT uses Direct prompts, with loss restricted to the answer JSON and end-of-sequence tokens. LoRA targets attention and feed-forward projections with rank 8, scaling factor 16, and zero dropout. We train for two epochs at a learning rate of 10^{-4}, with micro-batch size 1 and eight gradient accumulation steps. Training uses BF16, gradient checkpointing, and a 16,384-token sequence limit. Checkpoints at epochs 1 and 2 are evaluated under Direct and CoT on the same held-out queries.

The two-epoch results in Table 7 highlight three findings:

- **Learning both revision and preservation.** Direct-format SFT substantially improves BU and BM accuracy on *On CF-Path* queries.
- **Transfer to CoT prompting.** These gains carry over to CoT evaluation despite training exclusively on answer-only targets, without supervised reasoning traces.
- **Uneven gains on Off CF-Path queries.** Direct performance also improves outside counterfactual propagation, although accuracy remains below On CF-Path performance.

We also present each model's CoT responses before and after two SFT epochs on a shared counterfactual instance. The examples compare answer correctness and the consistency of cited support.

Table 7: SFT accuracy (%) and changes from Before (pp), by area and evaluation type.

		Direct			CoT		
Model	Metric	Before	1 ep.	2 ep.	Before	1 ep.	2 ep.
(a) CF-governed	BU: N=1499; BM: N=1105						
Qwen3-4B	BU	55.9	88.8 (+32.9)	91.3 (+35.4)	60.6	76.7 (+16.1)	83.9 (+23.2)
	BM	68.0	93.8 (+25.8)	96.7 (+28.8)	62.2	78.7 (+16.6)	83.8 (+21.6)
Qwen3-14B	BU	56.8	88.2 (+31.4)	92.9 (+36.1)	65.3	85.9 (+20.5)	85.8 (+20.5)
	BM	71.4	95.8 (+24.4)	96.4 (+25.0)	69.5	88.2 (+18.7)	89.3 (+19.8)
Qwen3-32B	BU	51.0	85.0 (+34.0)	92.3 (+41.4)	65.3	87.5 (+22.1)	90.8 (+25.5)
	BM	69.5	92.7 (+23.2)	96.1 (+26.6)	73.8	90.3 (+16.6)	91.5 (+17.7)
Gemma3-4B	BU	47.1	66.6 (+19.5)	71.8 (+24.7)	33.8	28.2 (-5.6)	36.6 (+2.8)
	BM	51.9	91.5 (+39.5)	92.6 (+40.6)	37.5	36.0 (-1.4)	49.2 (+11.8)
Gemma3-12B	BU	55.4	87.0 (+31.6)	92.3 (+37.0)	50.2	91.6 (+41.4)	93.7 (+43.4)
	BM	70.3	94.8 (+24.4)	96.6 (+26.2)	58.3	94.3 (+36.0)	94.4 (+36.1)
Gemma3-27B	BU	63.1	93.3 (+30.2)	96.5 (+33.4)	73.2	93.9 (+20.7)	92.6 (+19.4)
	BM	86.1	94.8 (+8.8)	97.7 (+11.7)	79.5	95.6 (+16.0)	92.9 (+13.4)
Llama3.1-8B	BU	52.4	79.2 (+26.8)	92.8 (+40.4)	54.6	57.4 (+2.8)	90.3 (+35.7)
	BM	55.4	89.8 (+34.4)	96.5 (+41.1)	60.5	62.8 (+2.4)	91.4 (+31.0)
Llama3.3-70B	BU	66.4	94.5 (+28.0)	97.0 (+30.6)	69.4	88.9 (+19.5)	90.5 (+21.1)
	BM	84.9	96.2 (+11.3)	97.2 (+12.3)	82.2	91.3 (+9.1)	92.6 (+10.4)
(b) Off-CF	BU: N=1800; BM: N=1351						
Qwen3-4B	BU	15.6	51.1 (+35.5)	50.0 (+34.4)	58.4	70.7 (+12.2)	72.3 (+13.9)
	BM	64.5	61.0 (-3.6)	74.3 (+9.8)	39.2	15.1 (-24.1)	20.5 (-18.7)
Qwen3-14B	BU	18.9	50.8 (+31.9)	53.9 (+35.1)	54.4	63.2 (+8.8)	67.8 (+13.4)
	BM	65.2	71.7 (+6.4)	76.2 (+11.0)	37.2	39.1 (+1.9)	40.3 (+3.0)
Qwen3-32B	BU	31.8	59.3 (+27.4)	53.9 (+22.1)	44.1	63.4 (+19.3)	68.6 (+24.6)
	BM	54.5	58.0 (+3.5)	77.8 (+23.3)	53.4	37.3 (-16.1)	34.9 (-18.5)
Gemma3-4B	BU	25.1	33.7 (+8.6)	32.2 (+7.1)	40.6	27.8 (-12.8)	34.2 (-6.4)
	BM	44.6	72.1 (+27.5)	75.7 (+31.1)	32.8	24.8 (-8.0)	31.8 (-1.0)
Gemma3-12B	BU	18.9	47.3 (+28.3)	52.0 (+33.1)	39.3	27.0 (-12.3)	36.3 (-3.0)
	BM	61.7	64.9 (+3.3)	68.6 (+7.0)	43.6	51.5 (+7.9)	57.8 (+14.2)
Gemma3-27B	BU	16.2	63.0 (+46.8)	62.6 (+46.4)	39.8	61.4 (+21.6)	68.4 (+28.6)
	BM	78.8	53.7 (-25.1)	73.6 (-5.2)	60.4	51.4 (-9.0)	51.2 (-9.2)
Llama3.1-8B	BU	15.4	28.2 (+12.8)	49.2 (+33.7)	15.1	16.8 (+1.7)	30.3 (+15.2)
	BM	52.5	76.7 (+24.2)	73.8 (+21.3)	59.0	49.7 (-9.3)	65.4 (+6.4)
Llama3.3-70B	BU	14.7	56.1 (+41.3)	67.4 (+52.7)	22.5	61.9 (+39.4)	69.2 (+46.7)
	BM	72.4	74.5 (+2.1)	72.6 (+0.2)	72.8	50.9 (-21.9)	36.6 (-36.3)

CoT · On CF-Path · Edit · SFT: before → 2 epochs

Input and intervention

Query	P51 (L5)
Initial facts	P00=F, P01=T, P02=T
Protected CF	empty → P32=T
Cut depth	1.0
Withdrawn rules	R13, R15, R27
Withdrawn facts	None

Rules into query (excerpt)

Rule	Implication	Order
R39	P40=F → P51=F	39
R40	P42=F → P51=F	40
R41	P41=T → P51=F	41
R42	P40=T → P51=F	42

Table 8: Model size and SFT on the shared On CF-Path input. Query: P51 at L5; Edit reference: F.

Model	Before SFT	After SFT
Qwen3-4B (S)	**value: U** (incorrect) The query P51 is determined by the rule R39: P40=F → P51=F. Initially, P40 is derived from P30, which is governed by P21 and P22. The cut at depth 1.0 removes rules that support the initial value of P11 and P12, including R13, R15, R27. . . . [excerpt]	**value: F** (correct) The protected counterfactual P32=T triggers CF-grounded support, which activates rule R32 (P32=T → P40=F). This leads to rule R39 (P40=F → P51=F). The rule R13, R15, and R27 are disabled, so P21=T and P32=F are not active, and thus do . . . [excerpt]
Qwen3-14B (M)	**value: U** (incorrect) The counterfactual sets P32=T, which triggers R28 (P32=F → P41=T) as CF-grounded support, but since P32=T, R28 is not applicable. However, R33 and R34 (P30=T → P41=T) are not triggered because P30 is undetermined. . . . [excerpt]	**value: F** (correct) The counterfactual sets P32=T, which triggers R32 to set P40=F. This leads to R39 to set P51=F. The cut disables R13, R15, and R27, but these do not affect the path from P32=T to P51=F. Thus, P51=F is the final value.
Qwen3-32B (L)	**value: F** (correct) The counterfactual sets P32=T as protected. This enables R30 and R32 to apply, but R32 is disabled by the cut. R30 sets P40=T. This triggers R39 and R42, which are both available. R39 (P40=F → P51=F) is not applicable, but R42 (P40=T → P51=F) applies and sets P51=F.. . . [excerpt]	**value: F** (correct) The counterfactual P32=T is protected. Rule R32 (P32=T → P40=F) is now CF-grounded and valid. Rule R32 leads to P40=F. Rule R39 (P40=F → P51=F) is now valid and CF-grounded, determining P51=F. No higher-priority supports contradict this. The cut disables R13, R15,. . . [excerpt]
Gemma3-4B (S)	**value: U** (incorrect) The counterfactual intervention disables R13, R15, and R27, which directly support P32=T. Since P32 is now protected as true, and no other supports for P51 exist, P51 is invalidated. The initial value of P51 was undetermined, and with no surviving supports, its value becomes U.	**value: F** (correct) The counterfactual P32=T, with cut depth 1.0, disables R13, R15, and R27. R13 and R15 are removed because they depend on P32=F. R27 is removed because it depends on P32=F. The initial value of P51 is U. R40 is now available, and its premise P42=F is true. . . . [excerpt]
Gemma3-12B (M)	**value: F** (correct) Initially, P51 is F due to P40=T (R37) and P42=F (R40). The counterfactual sets P32=T. The cut at depth 1.0 removes R13, R15, and R27. This prevents P32 from being F, so R30 and R31 are not applicable. P40 remains T (R30), and P42 remains F (R40). Therefore, P51 remains F due to R37 and R40.	**value: F** (correct) P40=F due to P32=T → P40=F, and P40=F → P51=F.
Gemma3-27B (L)	**value: F** (correct) Initially, P32=F. The counterfactual sets P32=T. The cut disables R13, R15, and R27. With P32=T, R30 applies, setting P40=T. R33 and R34 apply, setting P41=T. R42 applies, setting P51=F. Therefore, P51=F.	**value: F** (correct) The counterfactual sets P32=T, which enables rule R28. Rule R28 sets P40=F. Rule R39 then sets P51=F.

D.6 RL TUNING

GRPO uses the same training pool and LoRA configuration as SFT, but starts from the original instruction-tuned model rather than an SFT checkpoint. We sample four completions per CoT prompt at temperature 0.8 and top-p 1.0, with an 8,192-token completion limit. Reward is 1 for a format-compliant CoT JSON response with the correct final value and 0 otherwise; explanation quality is not scored. We use group-normalized rewards, a learning rate of 5×10^{-6}, a KL coefficient of 0.04, a clipping parameter of 0.2, and eight completions per optimizer update. Training runs for one epoch, with midpoint and final checkpoints evaluated under greedy decoding. SFT and GRPO use seed 0 and fixed checkpoint schedules without test-based selection; their training budgets are not compute-matched.

Table 9 shows two main patterns after one epoch:

- **On CF-Path.** GRPO improves both BU and BM across all five models, with generally larger gains under CoT prompting.
- **Off CF-Path.** BM improves consistently under both prompting styles, whereas BU changes little or declines. Gains in preservation therefore do not consistently extend to revision.

Table 9: GRPO accuracy (%) and changes from Before (pp), by area and evaluation type.

		Direct			CoT		
Model	Metric	Before	0.5 ep.	1 ep.	Before	0.5 ep.	1 ep.
(a) On CF-Path	BU: N=1499; BM: N=1105						
Qwen3-4B	BU	55.9	55.8 (-0.1)	56.4 (+0.5)	60.6	63.6 (+2.9)	63.8 (+3.1)
	BM	68.0	68.6 (+0.6)	68.1 (+0.1)	62.2	65.9 (+3.7)	68.0 (+5.8)
Qwen3-14B	BU	56.8	56.8 (+0.1)	57.5 (+0.7)	65.3	67.2 (+1.9)	67.4 (+2.1)
	BM	71.4	72.7 (+1.3)	73.6 (+2.2)	69.5	70.3 (+0.8)	72.4 (+2.9)
Gemma3-4B	BU	47.1	49.1 (+2.0)	49.1 (+2.0)	33.8	41.1 (+7.3)	43.0 (+9.2)
	BM	51.9	52.9 (+1.0)	53.9 (+2.0)	37.5	50.2 (+12.8)	49.8 (+12.3)
Gemma3-12B	BU	55.4	57.6 (+2.3)	57.4 (+2.1)	50.2	54.5 (+4.3)	56.9 (+6.7)
	BM	70.3	72.7 (+2.4)	73.0 (+2.7)	58.3	63.2 (+4.9)	64.9 (+6.6)
Llama3.1-8B	BU	52.4	54.3 (+1.9)	56.1 (+3.7)	54.6	57.8 (+3.2)	56.8 (+2.1)
	BM	55.4	60.9 (+5.5)	68.6 (+13.2)	60.5	67.1 (+6.7)	71.5 (+11.0)
(b) Off CF-path	BU: N=1800; BM: N=1351						
Qwen3-4B	BU	15.6	15.5 (-0.1)	15.6 (-0.1)	58.4	58.1 (-0.3)	58.3 (-0.1)
	BM	64.5	64.8 (+0.3)	65.2 (+0.7)	39.2	44.9 (+5.8)	42.3 (+3.2)
Qwen3-14B	BU	18.9	17.7 (-1.2)	17.6 (-1.3)	54.4	53.9 (-0.4)	53.7 (-0.7)
	BM	65.2	66.3 (+1.1)	66.7 (+1.5)	37.2	40.3 (+3.1)	39.8 (+2.6)
Gemma3-4B	BU	25.1	20.2 (-4.9)	19.4 (-5.7)	40.6	31.8 (-8.8)	30.7 (-9.8)
	BM	44.6	48.6 (+3.9)	48.9 (+4.2)	32.8	43.0 (+10.2)	45.1 (+12.3)
Gemma3-12B	BU	18.9	17.1 (-1.9)	17.2 (-1.7)	39.3	35.1 (-4.3)	33.5 (-5.8)
	BM	61.7	63.7 (+2.0)	64.0 (+2.4)	43.6	52.2 (+8.6)	52.3 (+8.7)
Llama3.1-8B	BU	15.4	15.9 (+0.4)	16.1 (+0.6)	15.1	15.1 (+0.0)	15.3 (+0.2)
	BM	52.5	56.6 (+4.1)	62.3 (+9.8)	59.0	63.2 (+4.2)	63.6 (+4.6)

D.7 Inferred versus Supplied Belief States

We evaluate whether explicitly supplying initial beliefs and support histories helps LLMs make the required revisions and preservation decisions after a counterfactual intervention. Table 10 illustrates three input conditions. *Initial* supplies only source facts and ordered rules, requiring the model to infer the initial state. B_0 additionally supplies initial belief values, while $B_0 + \mathrm{Hist.}$ also provides ordered support histories. The supplied information describes the pre-intervention state and does not introduce additional source facts. Queries and reference answers remain fixed across input conditions.

Table 10: Initial-state input variants for the same example (excerpts).

Shared input (excerpt)
Initial facts: P00=F, P01=T, P02=T, P03=F
Rules (excerpt): R24 [order 24]: P23=F → P30=F; R25 [order 25]: P22=F → P30=F
R26 [order 26]: P20=T → P30=T; R27 [order 27]: P22=T → P30=T
R40 [order 40]: P30=T → P41=T; R41 [order 41]: P30=T → P41=T
R52 [order 52]: P41=T → P51=F; R53 [order 53]: P41=T → P51=T
Intervention: protected CF: P30=F; cut rules: R00, R01, R02, R14, R15, R26; cut facts: none; depth: 2.0
Query / method: P51 / Edit, CoT

Initial `initial_only`	B_0 `b0_given`	B_0 **+ history** `b0_history`
Input-view instruction (excerpt) Only initial source facts and rules are supplied. First construct the initial belief state from them, before applying the intervention. Process source facts in the supplied initial fact order, then process derived layers in increasing order and incoming rules in their supplied order.	**Input-view instruction (excerpt)** The supplied truth values describe the propositions before the intervention. Support histories are not supplied, and these values are not the final answer. Reason from these values together with the initial source facts and ordered rules. When support information is needed, derive it from those inputs; do not assume that a truth value alone identifies its support or history.	**Input-view instruction (excerpt)** The supplied truth values and ordered support histories describe the state before the intervention. They are not the final answer. Reason from these values and histories together with the initial source facts and rules. Use the supplied histories to identify existing supports and their order; do not treat derived values as additional source facts.
Initial-state values (excerpt) Not provided; construct the initial state from the initial inputs.	**Initial-state values (excerpt)** P20=T, P22=F P23=F, P30=T P41=T, P51=T	**Initial-state values (excerpt)** P20=T, P22=F P23=F, P30=T P41=T, P51=T
Ordered histories (excerpt) Not supplied.	**Ordered histories (excerpt)** Not supplied.	**Ordered histories (excerpt)** P30: R24: value=F, birth=13 priority=0, valid=true R25: value=F, birth=14 priority=0, valid=true R26: value=T, birth=15 priority=0, valid=true P41: R40: value=T, birth=21 priority=0, valid=true R41: value=T, birth=22 priority=0, valid=true P51: R52: value=F, birth=30 priority=0, valid=true R53: value=T, birth=31 priority=0, valid=true

Table 11 compares Rebuild and Edit on counterfactual-governed queries. Providing B_0 often improves BM accuracy without corresponding BU gains, and adding histories does not consistently improve either metric. For Qwen3-14B under Edit and Direct prompting, supplying values and histories raises one-hop BM accuracy from 67.8% to 85.9%, but lowers BU accuracy from 56.4% to 25.6% relative to Initial. Thus, access to initial beliefs and their support histories does not by itself ensure correct revision.

Table 11: Rebuild and Edit: CF-governed BU and BM standardized accuracy (%).

	Rebuild								Edit								Average	
	BU				BM				BU				BM				BM	BU
	Direct		CoT		Direct		CoT		Direct		CoT		Direct		CoT			
Input	h-1	h-2	h-1	h-2	h-1	h-2	h-1	h-2	h-1	h-2	h-1	h-2	h-1	h-2	h-1	h-2		
Qwen3-4B																		
Initial	53.6	58.7	65.3	**49.6**	66.1	69.0	66.3	48.5	52.6	58.5	71.6	**50.4**	65.4	67.1	70.7	50.9	63.0	57.5
B0	**60.0**	**63.1**	**70.3**	48.1	62.4	62.5	**71.0**	**61.9**	**60.5**	**60.9**	**72.3**	44.6	62.0	63.1	**82.5**	**70.8**	67.0	**60.0**
B0 + H	39.6	43.0	43.3	31.0	**75.6**	**77.9**	54.1	50.8	36.0	41.8	44.9	28.4	**81.3**	**83.8**	63.8	59.6	**68.4**	38.5
Qwen3-14B																		
Initial	**50.8**	**55.4**	**76.5**	**55.2**	62.9	66.2	79.0	64.9	**56.4**	**60.6**	**76.9**	**53.8**	67.8	72.1	76.7	58.6	68.5	**60.7**
B0	43.3	50.9	74.7	51.1	73.7	**71.4**	**84.0**	**79.8**	48.2	57.6	70.6	50.9	74.7	69.6	**86.6**	**76.3**	**77.0**	55.9
B0 + H	26.5	33.4	41.8	39.0	**79.5**	70.3	61.7	72.0	25.6	37.7	37.0	38.2	**85.9**	**74.9**	61.8	68.7	71.8	34.9
Qwen3-32B																		
Initial	**50.5**	**52.3**	**80.5**	**57.3**	61.0	**66.8**	80.1	66.4	**52.1**	**49.0**	**74.4**	**58.8**	66.8	**69.3**	78.1	67.5	69.5	**59.3**
B0	35.5	38.4	70.9	45.8	**71.0**	63.3	**85.3**	**79.2**	30.4	38.3	57.3	42.2	**79.7**	67.0	**83.3**	**76.6**	**75.7**	44.8
B0 + H	28.0	27.7	35.9	33.8	67.5	63.3	63.4	71.4	33.8	32.1	25.7	29.0	59.3	42.1	51.3	61.1	59.9	30.8
Gemma3-4B																		
Initial	47.5	47.4	38.3	**38.6**	50.1	49.4	40.1	41.4	49.4	49.5	**34.0**	**32.6**	50.8	52.4	38.4	35.7	44.8	**42.2**
B0	49.1	**48.7**	**41.7**	36.1	**56.4**	**58.7**	**50.4**	**51.5**	**50.1**	**50.7**	27.5	27.4	53.6	53.5	42.8	41.5	**51.1**	41.4
B0 + H	**50.2**	47.2	25.1	19.7	54.9	57.3	37.7	36.6	50.0	50.4	31.4	23.9	**54.1**	**55.8**	**52.3**	**45.4**	49.3	37.2
Gemma3-12B																		
Initial	**41.8**	**41.9**	**59.4**	**49.9**	54.7	52.7	64.0	57.6	**54.8**	**57.8**	**55.6**	**48.1**	69.4	68.2	59.2	54.6	60.1	**51.2**
B0	31.5	26.1	54.4	37.1	**77.4**	**81.2**	**80.2**	**76.4**	38.9	31.0	46.7	32.3	**78.1**	**78.7**	**75.8**	**74.0**	**77.7**	37.2
B0 + H	37.2	37.2	44.0	33.7	72.9	74.4	67.9	70.4	43.5	40.5	34.9	28.0	70.9	72.4	59.8	63.6	69.0	37.4
Gemma3-27B																		
Initial	**62.9**	**67.0**	74.1	**66.7**	87.1	82.3	84.0	75.4	**62.2**	**67.0**	**82.8**	**65.7**	86.2	84.3	85.2	71.7	82.0	**68.5**
B0	43.1	33.5	**74.1**	46.6	**92.4**	**95.1**	**89.8**	**87.9**	38.4	30.1	75.2	48.3	**90.7**	**94.1**	**85.9**	**81.9**	**89.7**	48.7
B0 + H	22.1	25.8	68.2	40.4	89.7	87.7	85.7	80.8	20.3	25.6	56.6	39.1	88.1	85.3	76.0	73.5	83.3	37.3
Llama3.1-8B																		
Initial	**49.9**	**53.5**	**53.3**	**56.0**	55.1	60.8	54.7	58.7	**49.5**	**52.0**	**54.4**	**57.0**	53.1	58.6	57.8	62.5	57.7	**53.2**
B0	43.7	45.9	46.4	45.6	65.5	78.3	**72.2**	**77.9**	45.1	48.2	39.4	35.7	60.9	76.2	**63.1**	**65.8**	70.0	43.7
B0 + H	30.7	35.3	40.3	39.6	**81.0**	**89.3**	64.9	70.1	33.5	37.9	31.5	31.2	**86.4**	**91.5**	58.4	50.3	**74.0**	35.0
Llama3.3-70B																		
Initial	67.5	64.3	**72.2**	**64.7**	**88.4**	81.3	82.5	80.5	**68.8**	**65.3**	**75.4**	**66.2**	**87.6**	80.8	84.9	78.1	**83.0**	**68.0**
B0	**68.0**	**68.7**	65.3	59.0	76.8	70.4	**89.1**	**83.9**	64.2	64.1	67.7	59.5	82.8	75.1	**86.7**	**80.2**	80.6	64.6
B0 + H	58.1	47.8	47.9	44.1	71.9	**81.9**	81.2	78.2	47.0	36.6	48.9	42.6	81.3	**88.5**	82.5	79.5	80.6	46.6