FIHR-CQA: STATEMENT-LEVEL FULL INFERENCE AND BUDGETED WITNESS LEARNING ON HYPER-RELATIONAL GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

Complex query answering over hyper-relational knowledge graphs requires a model to combine several statements while keeping each qualifier attached to the correct statement. Existing query-embedding and graph-agent methods can rank answers, but they do not reliably show that a prediction depends on multiple distinct missing statements. This gap has three sources: alternative witnesses can reuse observed components, separately ranked statement candidates may disagree on shared bindings, and answer supervision does not tell the model which evidence should be acquired. To address these issues, we propose FIHR-CQA and HyWit for statement-level answer–witness inference. FIHR-CQA audits all reference witnesses and removes distinctness, projection, and qualifier shortcuts. HyWit then composes typed statement candidates with exact joins and directs additional requests to unresolved shared bindings under a fixed budget. Joint answer–evidence training and policy-matched acquisition labels distinguish candidate coverage from final witness selection. Experiments on shared WD50K-NFOL strict targets show improved joint recovery over representative query models and matched-backbone search controls, with consistent gains under held-out structures and measured cost settings.

1 INTRODUCTION

Complex query answering (CQA) is used to find entities that satisfy several relational conditions at once (Hamilton et al., 2018; Ren et al., 2020; Ren & Leskovec, 2020; Arakelyan et al., 2021). Hyper-relational knowledge graphs make this task more precise by attaching qualifiers to individual facts (Vrandečić & Krötzsch, 2014; Rosso et al., 2020; Galkin et al., 2020). Consider a query asking which work earned a person an award and came from a given country. The answer must bind the same work in both statements, and a qualifier such as forWork must belong to the award statement that supports the answer. Returning the work alone does not show that these statement-level conditions are satisfied.

Neural query models encode logical structure in continuous representations (Ren & Leskovec, 2020; Zhang et al., 2021; Chen et al., 2022). Neuro-symbolic methods combine learned scores with query execution (Arakelyan et al., 2021; Bai et al., 2023; Yin et al., 2024). Hyper-relational models such as StarE, StarQE, and NQE preserve qualifiers during completion or query answering (Galkin et al., 2020; Alivanistos et al., 2022; Luo et al., 2023c), while DHGE and related graph encoders improve statement representations (Luo et al., 2023a;b). Full-inference evaluation has also shown that a query should be rejected when an easier alternative reasoning tree can answer it (Gregucci et al., 2025). These lines of work ad-

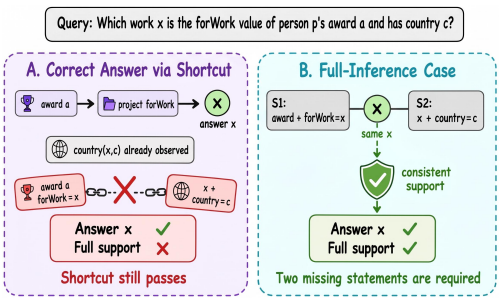

Figure 1: A correct answer requires compatible statements, not isolated role projections.

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.

dress important parts of the problem, but they do not by themselves establish that a returned answer needs several distinct missing statements with compatible qualifier ownership.

This limitation is easy to miss in a reified graph. Two missing role edges may belong to one statement, so counting missing edges can overstate the inference depth. A query may also have an observed component or a qualifier-free projection that reaches the same answer. Even when a target survives these checks, independently retrieved statement candidates can bind different entities to a shared variable. A decoder that keeps only the top candidates for each statement may therefore discard the combination that forms a valid witness. Finally, an answer loss rewards the entity but does not identify whether its selected evidence is supported. Extra computation then has no reliable signal for deciding which incomplete binding should be explored next.

We study statement-level full inference under these conditions. The task is to answer a qualified query and return a typed witness whose statements are executable under the query bindings. It raises three challenges. **First, benchmark targets must exclude all declared shortcut witnesses.** A sampled supporting path is insufficient when another reference witness uses an observed component, collapses statement variables, or satisfies a lossy role projection. **Second, witness construction must preserve statement identity and shared bindings.** Candidate rows that look plausible in isolation may not join into one supported assignment. **Third, limited acquisition must improve joint evidence rather than answer scores alone.** The model must decide which request is useful when the current candidate tables have unresolved separators and the reference graph cannot be consulted at test time.

To address these challenges, we introduce **FIHR-CQA**, a statement-level benchmark protocol, and **HyWit**, a budgeted answer–witness inference framework. FIHR-CQA checks alternative witnesses over the full reference snapshot, counts distinct missing statements, and tests a fixed family of qualifier and role projections. HyWit groups the roles of each statement into a typed candidate table and joins these tables through shared variables. When a separator has no compatible support, it requests more candidates for that binding and continues until the budget is exhausted. Training separates answer ranking from same-record evidence support, while policy labels compare requests under the same remaining budget. At inference, the framework uses only the observed graph and returns ranked answers with reconstructed witnesses.

We evaluate the protocol and the method on shared WD50K-NFOL strict targets. The study compares established qualified-query models, lossless reification, and matched-backbone search controls. It also tests component removal, held-out query structures, qualifier interventions, incomplete training support, and measured request cost. This design reports answer quality, joint witness recovery, candidate coverage, and the boundary between abstract request budgets and elapsed execution time.

Our main contributions are threefold:

- We introduce FIHR-CQA, which extends full-inference evaluation from missing representation edges to distinct statement support and publishes rejection certificates for alternative witnesses and declared projections.
- We develop HyWit, which combines exact separator-consistent joins with budgeted candidate acquisition for incomplete hyper-relational queries, returning both an answer and an executable typed witness.
- We formulate joint answer–evidence supervision and policy-matched request labels, and provide a cost-aware evaluation protocol that separates candidate coverage, witness selection, and execution overhead.

2 Related Work

Hyper-relational representation and query answering. Hyper-relational models retain the roles and qualifiers attached to a fact (Wen et al., 2016; Guan et al., 2019; Rosso et al., 2020; Liu et al., 2021). StarE (Galkin et al., 2020) propagates qualifier-aware messages. GRAN (Wang et al., 2021) encodes statement structure, while DHGE (Luo et al., 2023a) couples instance and ontology information. These representations support statement completion. StarQE (Alivanistos et al., 2022) and NQE (Luo et al., 2023c) extend prediction to qualified queries. A high answer score, however, does

not specify which compatible statement assignments support it. Our task evaluates that assignment explicitly.

Full inference and query relaxation. Binary CQA combines query embeddings, fuzzy operators, and learned graph execution (Hamilton et al., 2018; Ren et al., 2020; Ren & Leskovec, 2020; Zhu et al., 2022). Full-inference evaluation already checks alternative reasoning trees (Gregucci et al., 2025). FIHR-CQA retains this requirement and changes the counting unit from missing representation edges to distinct statements. It also tests a fixed set of lossy role projections. Relaxation-based counting (Brunink et al., 2026) motivates these controls. Excluding the declared reductions does not exclude every statistical shortcut.

Structured inference and learned exploration. CQD (Arakelyan et al., 2021), QTO (Bai et al., 2023), and FIT (Yin et al., 2024) combine learned scores with logical composition. Variable elimination and factor-graph messages provide exact inference under suitable structural assumptions (Dechter, 1999; Kschischang et al., 2001). MINERVA (Das et al., 2018) and Graph-R1 (Luo et al., 2026) instead guide graph exploration. HyWit uses established exact joins within each candidate pool. Its learned decision concerns which additional statement candidates to acquire when existing rows cannot form useful witnesses.

3 PRELIMINARIES

This section defines qualified statements, query witnesses, and the budgeted prediction task.

Graph views and statement identity. An HKG contains entities $\mathcal{E}$, relation labels $\mathcal{R}$, and indexed statements $f_i = ((s_i, r_i, o_i), Q_i)$, where $Q_i \subseteq \mathcal{R} \times \mathcal{E}$ is a finite set of qualifier pairs. Component relations $M(i, s_i, r_i, o_i)$ and $Q(i, r^q, v^q)$ preserve qualifier ownership. The observed graph $\mathcal{G}_{\text{obs}}$ is a component-wise subgraph of a finite reference snapshot $\mathcal{G}_{\text{ref}}$. Reference identifiers support auditing only; models predict local statement contents, not hidden identifiers. Repeated qualifier labels with different values are retained.

Query and witness. A Core query is a positive existential query with one free entity variable x:

$$q(x) = \bigvee_{b \in \mathcal{B}_q} \exists \mathbf{z}_b \, \exists \boldsymbol{\delta}_b \bigwedge_{\ell \in L_b} \ell, \qquad \ell \in \{M(\delta, u, r, v),\, Q(\delta, r^q, y)\}. \tag{1}$$

The branch set $\mathcal{B}_q$, entity variables $\mathbf{z}_b$, and statement variables $\boldsymbol{\delta}_b$ are finite. Relations are constants; entity positions contain constants or entity variables. Each statement variable has one main atom and zero or more qualifier atoms, and each branch contains a qualifier. A reference witness $w = (b, \beta, \iota)$ binds entities through β and statement variables through ι to reference records. Different statement variables may share a record: no implicit inequality is imposed. The witness set $\mathcal{W}(q, a; \mathcal{G}_{\text{ref}})$ contains assignments satisfying all atoms with $\beta(x) = a$; its nonemptiness defines the answer set $A_{\text{ref}}(q)$.

Core scope and outputs. Grouping query atoms by statement variable produces entity-variable scopes $X_{b,j}$. Core requires this grouped hypergraph to admit a join tree in which the factors containing any variable form a connected subtree. Each branch connects an anchor to the answer. Empty scopes encode Boolean constraints. Given the query q, observed graph $\mathcal{G}_{\text{obs}}$, and positive work budget B, the task returns ranked entities with local typed witnesses, or abstains when no complete candidate witness exists. Negation, unrestricted cyclic queries, and natural-language parsing are outside Core.

4 METHODOLOGY

In this section, we introduce HyWit. As shown in Figure 2, FIHR-CQA first identifies evaluation targets through a statement-level audit. HyWit then composes typed candidates and allocates additional proposals to missing shared bindings. Reference facts supply offline labels, never test-time inputs.

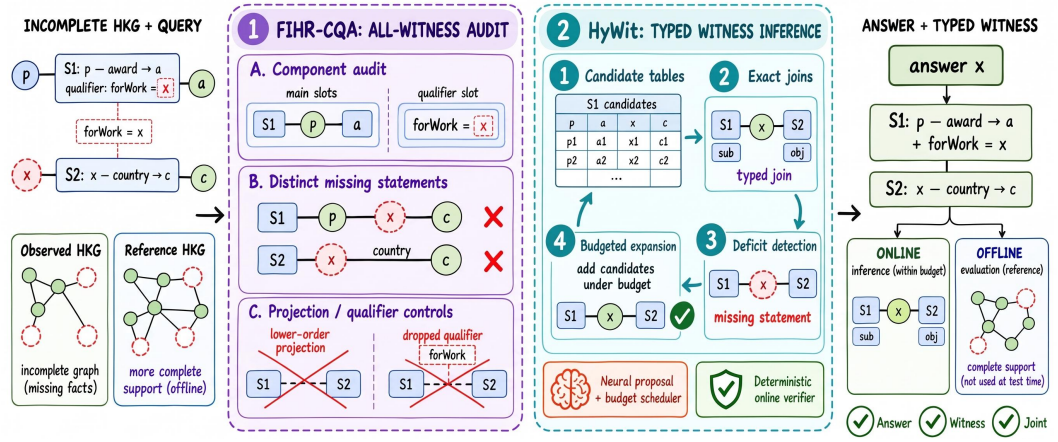

Figure 2: FIHR-CQA fixes evaluation conditions without supplying test-time evidence. HyWit expands statement tables and recomputes exact joins. Reference replay is restricted to training and offline evaluation.

4.1 Statement-Level Inference Audit

All-witness component audit. For branch atom ℓ, the restricted query $q^{\mathrm{obs}}_{b,\ell}$ evaluates that atom in $\mathcal{G}_{\mathrm{obs}}$ and all other atoms in $\mathcal{G}_{\mathrm{ref}}$. Its answers expose a reference witness using an observed component. The unsafe answer set is $U(q) = \bigcup_{b\in\mathcal{B}_q} \bigcup_{\ell\in L_b} \mathrm{Ans}(q^{\mathrm{obs}}_{b,\ell})$. An answer outside $U(q)$ has no such witness. Main-only and qualifier-only unions give diagnostic P-FI and Q-FI labels. Core requires both. This exact test covers alternative witnesses, not only a sampled path. A timeout yields an unresolved instance.

Distinct missing statements. For a reference witness w, the missing-statement count is $d(w) = |\{\iota(\delta) : \delta \in \boldsymbol{\delta}_b,\, M_{\iota(\delta)} \notin M^{\mathrm{obs}}\}|$. It counts records, not variable names or reified edges. Core requires $\min_{w\in\mathcal{W}(q,a;\mathcal{G}_{\mathrm{ref}})} d(w) \geq 2$. After the component audit, this condition has a simple exact test. For each branch, identify all statement variables with one variable to obtain its diagonal-collapse query $q^{=}_b$. Reject answers in $D(q) = \bigcup_b \mathrm{Ans}(q^{=}_b; \mathcal{G}_{\mathrm{ref}})$. Every surviving witness uses only missing main atoms, so fewer than two distinct missing records means that a diagonal-collapse query succeeds. The proof and the self-join counterexample are provided in Appendix C.

Projection and qualifier controls. A fixed grammar Γ forms qualifier-free, connected atom-deletion, and proper role-projection queries. Role projections retain selected entity positions but omit statement identifiers, losing common-statement constraints. All views use only $\mathcal{G}_{\mathrm{obs}}$. Their answer union is $P_\Gamma(q) = \bigcup_{\pi\in\Gamma(q)} \mathrm{Ans}(\pi(q); \pi(\mathcal{G}_{\mathrm{obs}}))$. A qualifier is non-vacuous when deleting its atom from the full query strictly enlarges the reference answer set. With $N(q)$ indicating that every qualifier passes this test, the strict target set is

$$A_{\mathrm{strict}}(q) = \begin{cases} A_{\mathrm{ref}}(q) \setminus \big(U(q) \cup D(q) \cup P_\Gamma(q)\big), & N(q) = 1, \\ \varnothing, & N(q) = 0. \end{cases} \tag{2}$$

Sequential rejection counts measure the added exclusions, but do not separate inference difficulty from changes in the retained population. Full details of the finite grammar and its rejection certificates are provided in Appendix B. The guarantee concerns this snapshot and grammar, not every shortcut or a model's internal reasoning.

Separation from binary full inference. Lossless reification preserves answers and witnesses, but several missing role edges can belong to one record. FIHR-CQA counts distinct records and tests lossy projections. Whole-statement removal makes P-FI and Q-FI identical, requiring separate component-mask diagnostics. Core groups identical main triples before splitting. The information-access contract for raw-split audits is given in Appendix A.

4.2 Separator-Consistent Witness Inference

Persistent candidate streams. For branch b, factor j groups one statement's atoms into scope $X_{b,j}$. Each candidate row binds these variables and retains the main triple and required qualifiers. A masked-role model proposes assignments with constants and separator bindings clamped. Decoding reuses repeated-variable bindings and fixes beam width, role order, and parameters at evaluation.

Requests resume persistent streams indexed by branch, factor, and conditioning assignment. Cursors retain decoding progress, so later requests do not rescore earlier prefixes. Discarded or duplicate proposals consume work; an empty response does not imply exhaustion. Appendix D specifies decoding and accounting.

Exact composition. Let C_j denote the acquired rows of factor j, and let the statement scorer $F_\theta(t_j, q)$ assign a real logit to row t_j. With m_b factors in branch b, its fixed potential is $\phi_j(t_j, q) = \log \sigma(F_\theta(t_j, q))/m_b$. For the set of compatible candidate witnesses $\mathcal{W}(C, q)$, define

$$S_\theta(a; C, q) = \max_{\substack{w \in \mathcal{W}(C,q) \\ \beta(x)=a}} \sum_{j \in b(w)} \phi_j(t_j, q). \tag{3}$$

The witness score $S_\theta(w, q)$ is the sum in the answer-score definition. Compatibility requires the query constants, role ownership, and every shared entity binding to agree. An empty maximum is $-\infty$. Branch averaging fixes the scoring convention; it does not make scores invariant to every equivalent query decomposition.

For a join-tree edge $i{-}j$, let $S_{ij} = X_i \cap X_j$ be its separator and $\mathcal{N}(i)$ the neighbors of factor i. The message for separator assignment u is

$$m_{i\to j}(u) = \max_{\substack{t_i \in C_i \\ t_i|_{S_{ij}} = u}} \left[\phi_i(t_i, q) + \sum_{k \in \mathcal{N}(i)\setminus\{j\}} m_{k\to i}(t_i|_{S_{ki}}) \right]. \tag{4}$$

A root containing answer variable x computes each answer score. Backpointers reconstruct a complete witness. Running intersection makes this computation exact within the current tables. It does not establish proposal completeness or factual correctness. The proof is provided in Appendix C.

Recursive deficit resolution. The deficit set for directed edge $i \to j$ is $\mathcal{D}_{i\to j} = \{u : m_{i\to j}(u) > -\infty,\ m_{j\to i}(u) = -\infty\}$. For each deficit, the resolver requests rows at receiving factor j. Existing rows matching u can induce further requests at children whose messages are absent. Traversal excludes the parent edge and stops at a fixed cap without declaring unvisited bindings impossible. Both local and downstream requests remain available.

Wildcard streams introduce bindings missing on both sides. A round-robin wildcard step occurs every r decisions. Nonexhausted requests reserve at most k score units within the remaining budget. Acquired tables grow monotonically, followed by message updates. Appendix D gives the complete transition.

4.3 Budget-Aware Witness Learning

Joint answer and evidence supervision. Training submasks hide observed records from the proposer and encoder. The unmasked training snapshot supplies answer labels and same-record witness support, not claims of real-world truth.

For a nonempty training answer set A_{tr}, define the best supported score $P_a = \max_{w \in \mathcal{W}_a^+} S_\theta(w, q)$, the best unsupported score for the *same* answer $N_a = \max_{w \in \mathcal{W}_a^-} S_\theta(w, q)$, and the best reference-incorrect answer score $R^- = \max_{e \notin A_{\mathrm{tr}}} S_\theta(e; C, q)$. For positive margins γ_a, γ_w, optimize

$$\begin{aligned} \mathcal{L}_{\mathrm{ans}} &= \frac{1}{|A_{\mathrm{tr}}|} \sum_{a \in A_{\mathrm{tr}}} [\gamma_a + R^- - P_a]_+, \\ \mathcal{L}_{\mathrm{ev}} &= \frac{1}{|A_{\mathrm{tr}}|} \sum_{a \in A_{\mathrm{tr}}} [\gamma_w + N_a - P_a]_+, \\ \mathcal{L}_{\mathrm{score}} &= \mathcal{L}_{\mathrm{ans}} + \lambda_w \mathcal{L}_{\mathrm{ev}} + \lambda_c \mathcal{L}_{\mathrm{comp}}. \end{aligned} \tag{5}$$

The completion loss $\mathcal{L}_{\text{comp}}$ retains observed-record training; nonnegative weights are validation-selected. Missing positive witnesses are injected only during scorer training, and empty negative sets contribute zero. Two-state messages compute supported and unsupported maxima without enumerating witnesses. The answer term separates entities; the evidence term separates derivations of the same entity. Details are provided in Appendix E.

Policy-matched acquisition labels. Freeze the proposer and scorer. The state h includes candidate tables, messages, cursors, traversal state, wildcard phase, and remaining budget. At iteration t, a mixture of frozen policy π_t and exploration collects states. Compare forcing affordable action α before π_t against following π_t immediately, using the same remaining allowance H. The forced request consumes work and advances the scheduling state.

For endpoint state h', let $r_{h'}(a)$ be the filtered rank of training answer a and $v_{h'}(a)$ indicate that its selected witness is reference-supported. Define terminal utility $U(h') = |A_{\text{tr}}|^{-1} \sum_a v_{h'}(a)/r_{h'}(a)$. The paired label is

$$\begin{aligned} y_t(h,\alpha) &= U(\mathcal{R}(h,\alpha;\pi_t,H)) - U(\mathcal{R}(h;\pi_t,H)), \\ \mathcal{L}_{\text{policy}} &= \mathbb{E}\big[(g_\psi(h,\alpha) - y_t(h,\alpha))^2 + \lambda_p (d_\psi(h,\alpha) - p(h,\alpha))^2\big]. \end{aligned} \tag{6}$$

The auxiliary target $p(h,\alpha)$ is the training-supported fraction of new unique rows, or zero for an empty response. Head d_ψ shares features with utility head g_ψ, but only g_ψ selects actions. Online features exclude support labels.

Rollouts include the forced action's cost without further immediate-cost normalization. Each iteration relabels retained states under its frozen continuation; validation selects an iterate, including the initial heuristic. Exact advantages admit a conditional improvement bound, not a guarantee for learned estimates. Full details are provided in Appendices C and E.

Execution and guarantees. Inference stops when no nonexhausted request is affordable. Positive request costs ensure termination. Candidate checks and exact joins guarantee structural validity, not reference support. With fixed potentials, growing tables preserve best scores but not ranks. Proofs are provided in Appendix C.

5 Experiments

This section presents the experimental setup, results, and analysis. We answer the following research questions (RQs): **RQ1:** Does HyWit improve joint answer–witness recovery? **RQ2:** Which acquisition and supervision components account for the gain? **RQ3:** What does statement-level auditing exclude beyond component tests? **RQ4:** Does acquisition transfer to held-out query structures? **RQ5:** How do qualifier changes and incomplete supervision affect recovery? **RQ6:** Do quality gains persist when execution cost is measured?

5.1 Experimental Setup

Dataset and evaluation population. We audit WD50K-NFOL (Luo et al., 2023c), starting from 17,920 query–answer pairs drawn from 4,850 queries. Sequential component, distinctness, projection, and non-vacuity tests retain 6,675 strict targets. Every main-result method uses this same target set and the same observed graph. Reference facts supply evaluation labels, not inference-time candidates or policy features. The audit counts targets rather than queries, since answers to the same query can have different inference requirements. Full task and information-access details are provided in Appendix A.

Baselines. We compare NQE (Luo et al., 2023c), StarQE (Alivanistos et al., 2022), and lossless-reification adaptations of CQD (Arakelyan et al., 2021) and QTO (Bai et al., 2023). For models without native witnesses, a reference-blind extractor supplies evidence without changing their answer ranks; its execution cost is charged. Static acquisition, a deficit heuristic, beam search with width five, and best-first search share HyWit's proposer and scorer backbones. These controls isolate acquisition from representation quality. Lossless reification preserves statement ownership and is not treated as a lossy shortcut. Adapter contracts are given in Appendix G.

Table 1: Results on the shared WD50K-NFOL strict targets. Values are mean ± standard deviation across ten paired seeds. Search controls share the proposer and scorer backbones.

Method	MRR	Hits@1	JointHit@1	Coverage
NQE + Extractor	46.2 ± 0.4	27.5 ± 0.3	16.2 ± 0.3	40.1 ± 0.5
StarQE + Extractor	46.8 ± 0.5	28.1 ± 0.4	17.5 ± 0.4	41.5 ± 0.6
CQD (lossless reified)	45.9 ± 0.6	26.8 ± 0.5	15.3 ± 0.3	38.6 ± 0.7
QTO (lossless reified)	47.6 ± 0.4	28.7 ± 0.3	18.2 ± 0.4	42.1 ± 0.5
Static acquisition	46.5 ± 0.3	27.9 ± 0.4	17.0 ± 0.2	40.4 ± 0.4
Deficit heuristic	47.1 ± 0.4	28.6 ± 0.3	18.3 ± 0.4	41.8 ± 0.5
Beam search ($w = 5$)	47.5 ± 0.3	29.1 ± 0.4	19.1 ± 0.3	42.6 ± 0.4
Best-first search	47.8 ± 0.4	29.4 ± 0.3	19.7 ± 0.3	43.1 ± 0.5
HyWit	$\mathbf{48.7 \pm 0.4}$	$\mathbf{30.2 \pm 0.3}$	$\mathbf{21.4 \pm 0.4}$	$\mathbf{44.8 \pm 0.5}$

Table 2: Component ablations on WD50K-NFOL. The fixed-pool evidence ablation isolates scoring on unchanged candidates; the other learned variants are retrained.

Variant	MRR	Hits@1	JointHit@1	Coverage
Full HyWit	**48.7**	**30.2**	**21.4**	**44.8**
Fixed-policy labels	48.2	29.5	20.3	43.7
No recursive requests	47.9	29.1	19.4	41.6
No evidence loss (fixed pool)	48.7	30.2	18.5	44.8
No evidence loss (retrained)	47.5	28.8	17.8	43.5
No answer loss	46.1	26.9	18.9	44.3
No auxiliary supervision	48.3	29.7	20.1	43.4
Static acquisition	46.5	27.9	17.0	40.4

Metrics and replication. MRR and Hits@1 measure filtered answer ranking. JointHit@1 requires a correct rank-one answer and a reference-supported selected witness. Coverage measures whether the acquired pool contains any supported target witness, irrespective of its final rank. All four metrics are percentages. The main comparison reports means and standard deviations over ten paired random seeds; other studies report aggregate scores. We distinguish abstract request budgets from measured median end-to-end latency. Appendix F defines the metrics and their denominators. We do not infer statistical significance from aggregate standard deviations.

5.2 Main Results (RQ1)

As shown in Table 1, HyWit achieves the highest scores on all four metrics. Against best-first search, the strongest search control, it improves JointHit@1 from 19.7% to 21.4% and coverage from 43.1% to 44.8%. The corresponding Hits@1 gain is smaller, from 29.4% to 30.2%. Thus, the improvement is not confined to answer ranking: more target witnesses enter the pool, and more correct answers retain valid evidence.

Among the published-model adaptations, QTO gives the strongest joint recovery at 18.2%. Its gap to HyWit is 3.2 percentage points under the common target and extraction protocol. These are adapted strict-target scores, not replacements for the methods' published benchmark results. Coverage remains substantially higher than JointHit@1 for every method. HyWit's 44.8% coverage and 21.4% joint recovery show that acquiring a supported witness does not ensure that the system selects it with the correct top-ranked answer.

5.3 Ablation Study (RQ2)

As shown in Table 2, candidate acquisition and witness selection have distinct effects. Removing recursive requests reduces coverage from 44.8% to 41.6% and joint recovery from 21.4% to 19.4%, consistent with unresolved bindings limiting candidate support. Training the controller with fixed-policy labels yields 20.3% joint recovery, below the full policy-matched procedure.

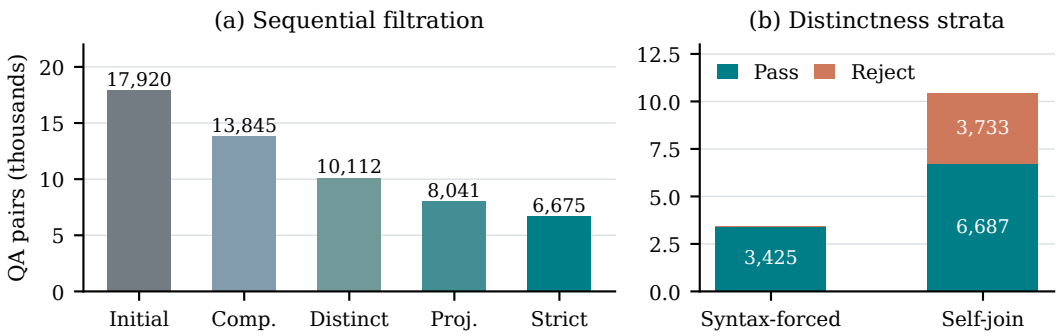

Figure 3: Sequential target filtration and distinctness stratification. Syntax-forced targets pass automatically; self-join-compatible targets require graph execution. Counts are query–answer pairs.

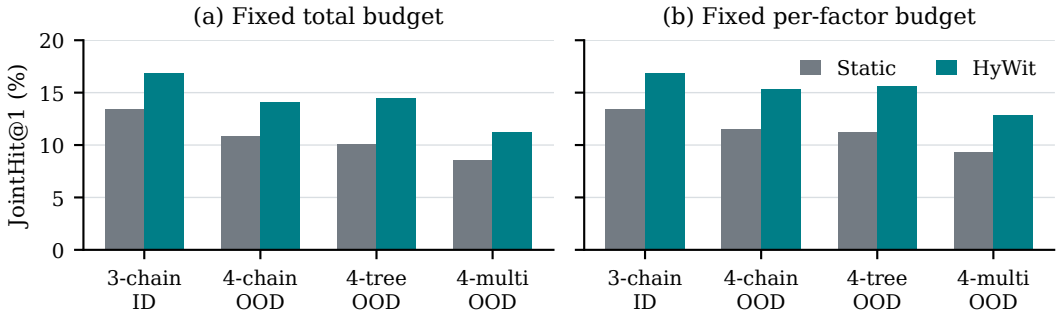

Figure 4: Joint recovery on in-distribution (ID) and held-out (OOD) structures. The two budget protocols separate a fixed total allowance from an allowance scaled by factor count. Multi denotes a multi-variable separator.

Evidence supervision has a distinct effect. On the fixed pool, removing its loss leaves coverage and rounded answer metrics unchanged but lowers JointHit@1 to 18.5%. The end-to-end retrained variant falls further to 17.8%, with coverage decreasing to 43.5%. The first comparison isolates evidence selection; the second includes changes to acquisition and scoring. Conversely, removing answer supervision lowers Hits@1 to 26.9% despite retaining 44.3% coverage. Auxiliary supervision contributes a smaller joint gain, from 20.1% to 21.4%. Together, these comparisons support complementary roles for acquisition, answer ranking, and evidence selection rather than treating any one component as sufficient.

5.4 Statement-Level Audit (RQ3)

As shown in Figure 3, the component test removes 4,075 targets, leaving 13,845. Of these, 3,425 have syntax that precludes collapsing all statement variables to one record. The remaining 10,420 allow such a collapse and require the diagonal query test $D(q)$. This test rejects 3,733 and retains 6,687, giving 10,112 distinctness-qualified targets in total. Missing components alone therefore do not establish that an answer requires multiple distinct missing records.

Projection controls remove another 2,071 targets, and non-vacuity removes 1,366. The final 6,675 targets constitute 37.2% of the initial population. These counts establish that the added controls are active on WD50K-NFOL; they do not establish the absence of shortcuts outside the declared grammar. The counts are also order-dependent: each stage receives only the preceding stage's survivors. Appendix B provides the rejection conditions.

5.5 Structural Transfer (RQ4)

As shown in Figure 4, HyWit's recovery advantage extends to held-out four-factor structures. At fixed total budget, HyWit exceeds Static by 3.3 points on chains and 4.4 points on branching trees.

Table 3: Qualifier intervention and support masking; answers and coverage are recomputed against the reference graph. J@1 denotes JointHit@1.

	J@1		Coverage	
Condition	Static	HyWit	Static	HyWit
Standard queries	17.0	21.4	40.4	44.8
Swapped qualifiers	5.2	7.1	14.3	16.5
Mask 10%	15.6	19.3	39.0	42.2
Mask 20%	13.9	17.1	37.1	39.4
Mask 40%	10.7	13.4	31.8	33.7

The four-factor comparisons hold query length constant, although topology and target composition can still differ.

Multi-variable separators are harder for both methods: joint recovery is 8.5% for Static and 11.2% for HyWit. With a fixed per-factor allowance, these scores increase to 9.3% and 12.8%. Additional proposals therefore help, but do not remove the difficulty of matching several shared bindings. HyWit's advantage persists under both protocols, supporting transfer beyond the training topology without claiming unrestricted structural generalization.

5.6 Semantic Intervention and Missing Supervision (RQ5)

As shown in Table 3, swapping qualifiers reduces coverage for both methods. Answers and support are recomputed for the modified queries. The altered qualifier combinations rarely co-occur on single records, leaving fewer recoverable witnesses. HyWit retains a joint advantage, but this distribution-changing intervention alone does not establish that its predictions depend causally on qualifier ownership.

The masking rows in Table 3 isolate supervision incompleteness while retaining the evaluation reference. At 40% masking, JointHit@1 falls from 21.4% to 13.4% for HyWit and from 17.0% to 10.7% for Static. HyWit's coverage declines from 44.8% to 33.7%, narrowing its coverage advantage from 4.4 to 1.9 points. The method remains useful relative to Static, but its absolute performance depends on available support. Holding the evaluator fixed prevents label changes from being mistaken for robustness; it does not eliminate false-negative supervision during training.

5.7 Budget and Execution Cost (RQ6)

As shown in Figure 5, request efficiency and elapsed time differ. At the base budget, HyWit reaches 21.4% joint recovery in 168 ms, compared with Static's 17.0% in 135 ms; equal request allowances therefore do not imply equal latency. From $2\times$ to $4\times$, HyWit's joint recovery rises only from 24.1% to 25.3%, while latency rises from 315 to 590 ms.

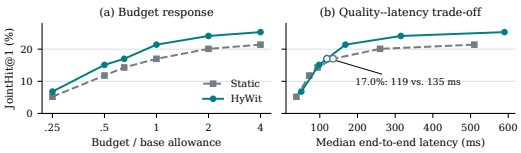

Figure 5: Budget response and measured quality–latency trade-off.

At matched quality, HyWit reaches 17.0% at the measured $0.65\times$ setting in 119 ms, compared with Static's 135 ms at $1\times$. It reaches 21.4% in 168 ms, whereas Static requires 510 ms at $4\times$. These operating points support an online quality–latency advantage, not a universal speedup; offline training cost is not included.

6 Conclusion

In this paper, we introduce FIHR-CQA and HyWit for statement-level answer–witness inference. The combined audit, exact composition, and budgeted acquisition improve joint recovery on shared WD50K-NFOL targets while exposing the method's evidence and latency limits.

AI USE STATEMENT

AI tools assisted with manuscript revision, mathematical exposition, computational-study code, and figure generation. The authors are responsible for checking the claims, references, implementation, and reported evidence.

REFERENCES

Dimitrios Alivanistos, Max Berrendorf, Michael Cochez, and Mikhail Galkin. Query embedding on hyper-relational knowledge graphs. In *International Conference on Learning Representations*, 2022.

Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. Complex query answering with neural link predictors. In *International Conference on Learning Representations*, 2021.

Yushi Bai, Xin Lv, Juanzi Li, and Lei Hou. Answering complex logical queries on knowledge graphs via query computation tree optimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 1472–1491, 2023.

Yannick Brunink, Daniel Daza, Yunjie He, and Michael Cochez. Counting still counts: Understanding neural complex query answering through query relaxation. *Transactions on Machine Learning Research*, 2026. URL https://arxiv.org/abs/2511.22565. Accepted version; arXiv:2511.22565v2.

Xuelu Chen, Ziniu Hu, and Yizhou Sun. Fuzzy logic based logical query answering on knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 3939–3948, 2022. doi: 10.1609/aaai.v36i4.20310.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *International Conference on Learning Representations*, 2018.

Rina Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113 (1–2):41–85, 1999. doi: 10.1016/S0004-3702(99)00059-4.

Mikhail Galkin, Priyansh Trivedi, Gaurav Maheshwari, Ricardo Usbeck, and Jens Lehmann. Message passing for hyper-relational knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7346–7359, 2020. doi: 10.18653/v1/2020.emnlp-main.596.

Cosimo Gregucci, Bo Xiong, Daniel Hernández, Lorenzo Loconte, Pasquale Minervini, Steffen Staab, and Antonio Vergari. Is complex query answering really complex? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 20399–20428, 2025.

Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. Link prediction on n-ary relational data. In *Proceedings of The Web Conference 2019*, pp. 583–593, 2019. doi: 10.1145/3308558.3313414.

William L. Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. Embedding logical queries on knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 31, pp. 2030–2041, 2018.

Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001. doi: 10.1109/18.910572.

Yu Liu, Quanming Yao, and Yong Li. Role-aware modeling for n-ary relational knowledge bases. In *Proceedings of The Web Conference 2021*, pp. 2660–2671, 2021. doi: 10.1145/3442381.3449874.

Haoran Luo, Haihong E, Ling Tan, Gengxian Zhou, Tianyu Yao, and Kaiyang Wan. DHGE: Dual-view hyper-relational knowledge graph embedding for link prediction and entity typing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 6467–6474, 2023a. doi: 10.1609/aaai.v37i5.25795.

Haoran Luo, Haihong E, Yuhao Yang, Yikai Guo, Mingzhi Sun, Tianyu Yao, Zichen Tang, Kaiyang Wan, Meina Song, and Wei Lin. HAHE: Hierarchical attention for hyper-relational knowledge graphs in global and local level. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8095–8107, 2023b. doi: 10.18653/v1/2023.acl-long.450.

Haoran Luo, Haihong E, Yuhao Yang, Gengxian Zhou, Yikai Guo, Tianyu Yao, Zichen Tang, Xueyuan Lin, and Kaiyang Wan. NQE: N-ary query embedding for complex query answering over hyper-relational knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 4543–4551, 2023c. doi: 10.1609/aaai.v37i4.25576.

Haoran Luo, Haihong E, Guanting Chen, Qika Lin, Yikai Guo, Fangzhi Xu, Zemin Kuang, Meina Song, Xiaobao Wu, Yifan Zhu, and Anh Tuan Luu. Graph-R1: Towards agentic GraphRAG framework via end-to-end reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

Hongyu Ren and Jure Leskovec. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 33, pp. 19716–19726, 2020.

Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *International Conference on Learning Representations*, 2020.

Paolo Rosso, Dingqi Yang, and Philippe Cudré-Mauroux. Beyond triplets: Hyper-relational knowledge graph embedding for link prediction. In *Proceedings of The Web Conference 2020*, pp. 1885–1896, 2020. doi: 10.1145/3366423.3380257.

Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014. doi: 10.1145/2629489.

Quan Wang, Haifeng Wang, Yajuan Lyu, and Yong Zhu. Link prediction on n-ary relational facts: A graph-based approach. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 396–407, 2021. doi: 10.18653/v1/2021.findings-acl.35.

Jianfeng Wen, Jianxin Li, Yongyi Mao, Shini Chen, and Richong Zhang. On the representation and embedding of knowledge bases beyond binary relations. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 1300–1307, 2016.

Hang Yin, Zihao Wang, and Yangqiu Song. Rethinking complex queries on knowledge graphs with neural link predictors. In *International Conference on Learning Representations*, 2024.

Zhanqiu Zhang, Jie Wang, Jiajun Chen, Shuiwang Ji, and Feng Wu. ConE: Cone embeddings for multi-hop reasoning over knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 34, pp. 19172–19183, 2021.

Zhaocheng Zhu, Mikhail Galkin, Zuobai Zhang, and Jian Tang. Neural-symbolic models for logical queries on knowledge graphs. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 27454–27478, 2022.

A Task Contract and Information Access

The following sections give the information-access contract, exact audit rules, formal properties, inference and training procedures, and evaluation details. They distinguish properties of finite query execution from performance of the learned acquisition policy.

Core logic. All domains are finite. Each disjunct is a positive conjunctive query with one answer entity, fixed relation labels, one main atom per statement variable, and at least one qualifier atom. Shared entity variables denote equality. Different statement variables do not denote inequality. Duplicate atoms and alpha-equivalent duplicate branches are removed before auditing and inference. The grouped entity-variable scopes, rather than the ungrouped atom graph, must satisfy the running-intersection property. For disconnected Boolean constraints, an empty separator connects the factor to the join tree; a branch without a path from an anchor to the answer is outside Core.

Reference support. A local statement proposal contains a main triple and the qualifier pairs required by its query factor. Additional qualifiers on a reference record do not invalidate that proposal. Support requires all required pairs to occur on *one* record with that main triple. Two records containing different subsets cannot jointly support one proposal. Equal local contents are deduplicated, but two different records with the same main triple are not merged by unioning their qualifier sets. Such a union could create a statement that never existed.

Views and data separation. Let $G_0 \subseteq G_1 \subseteq G_2$ be nested, deduplicated snapshots with disjoint held-out record groups. The primary fixed-observation protocol gives every evaluated model G_0. Validation labels come from G_1, and test labels and filtered answer sets come from G_2. Final evaluation does not retrain on G_1; an expanded-observation experiment, if used, is reported separately. Training witness supervision uses only records in G_0. Artificial submasks $G_{\mathrm{sub}} \subset G_0$ create training inference tasks, with G_0 as their training reference. No record in $G_2 \setminus G_0$ supplies training positives, negatives, utility features, or rollout labels.

For each submask, a structural encoder reads G_{sub}, not the omitted edges. Shared parameters can have encountered a record in another training submask; this is legitimate training supervision and is disclosed. It is not equivalent to the record-disjoint validation/test setting. Held-out query structures and validation targets provide the model-selection check against this training mismatch.

Table 4: Allowed access. Validation labels select hyperparameters; they do not update model weights in the primary protocol.

Component	Observed graph	Training reference	Validation labels	Test reference
Statement encoder/proposer	yes	training only	no	no
Structured scorer training	yes	yes	no	no
Utility rollout generation	yes	yes	no	no
Hyperparameter selection	yes	no	yes	no
Test-time proposal/policy	yes	no	no	no
Benchmark audit/evaluation	yes	as needed	as needed	yes

Canonicalization and leakage. Exact duplicate records are removed. Records sharing an identical main triple are assigned to the same Core partition, without merging their qualifier sets. Entity aliases and inverse-equivalent relations are normalized only by public rules fixed before splitting. If those rules are unavailable, residual alias and inverse leakage are reported rather than assumed absent. Query identifiers, record order, raw filenames, and hidden record IDs are never model features. Component-mask diagnostics retain stable evaluator-side record identity; models receive only observable local handles and contents. Main-content reuse across distinct handles is reported as an additional diagnostic.

One task, two contributions to evaluation. The prediction task is to rank entities and return executable typed witnesses under incomplete observations. FIHR-CQA selects the evaluation pop-

ulation without becoming an input to the predictor. Its audit and HyWit's acquisition algorithm are independently testable. Audit validity alone does not establish an acquisition gain.

B Audit Grammar and Rejection Certificates

Published-query audit versus new benchmark. The WD50K-NFOL audit preserves its original records and query labels, then classifies which instances can be mapped to Core. Conversion errors, unsupported negation, cyclic grouped scopes, or unavailable witness provenance receive separate statuses. New WD50K and WikiPeople suites follow the normalized record-group split. Their scores cannot be presented as if they were the original published benchmark scores.

Component flags. Write U_{P} for the component-rejection union restricted to main atoms and U_{Q} for its qualifier-atom counterpart. For reference answers, P-FI means $a \notin U_{\mathrm{P}}$, Q-FI means $a \notin U_{\mathrm{Q}}$, and J-FI means both. A branch without qualifiers would make its Q-FI requirement vacuous, so such branches are excluded from Core. Qualifier non-vacuity is a query-level property: it does not prove that a specific retained answer would change if its qualifier changed.

Finite reduction grammar. The primary grammar is frozen before any test-model comparison. It contains the following transformations, applied branchwise and then unioned.

1. **Qualifier-free reduction.** Remove qualifier atoms, retaining only branches in which the target remains constrained and connected to an anchor.
2. **Connected atom deletion.** Enumerate proper subsets of the original atoms that retain the answer, an anchor-to-answer connection, and each retained qualifier's main atom. Discard branches with an unconstrained answer. Core query sizes bound this finite enumeration; the release records that bound.
3. **Proper role projections.** For each grouped statement pattern, index its subject, object, and qualifier-value positions by their roles. A view retains one or two entity positions and the main relation label, plus only the qualifier labels of retained qualifier positions. It existentially drops the other entity positions, their qualifier requirements, and the statement identifier. The retained subset must be a proper subset of the original entity positions. All values in one view row must come from a single observed record. Different view rows need not share a record identifier.

For the third family, replace each original statement factor by a nonempty selection of its eligible views, retaining the original equalities on surviving variables. Enumerate combinations that keep an anchor-to-answer connection and bind the target. The resulting query may be easier because discarded roles are not checked, or because separate projections can be joined using different source records. Every chosen view is evaluated solely on observed components; an unobserved subject or main relation cannot be filled from the reference snapshot. With repeated qualifier labels, each retained occurrence remains paired with its own value position; projection does not silently impose all-different values.

This grammar is finite because the original query and role-position sets are finite. Its exact-evaluation cost can nevertheless be exponential in query size. A resource limit must return *unresolved*, not infer cleanliness from a truncated enumeration. Unions of transformations are allowed only as listed in the released grammar; arbitrary learned reductions are outside its guarantee.

Non-vacuity test. For every canonical qualifier atom ℓ, evaluate the full query $q^{-\ell}$ with that atom removed. Accept the non-vacuity condition only if $\mathrm{Ans}(q^{-\ell}; \mathcal{G}_{\mathrm{ref}}) \supsetneq A_{\mathrm{ref}}(q)$. If the removed atom was the only occurrence of a variable, the finite entity domain provides its existential semantics; it does not invent an anchor. Equivalent duplicate branches are removed before this test. A different branch can render a qualifier redundant, which is correctly detected by comparing *full-query* answer sets.

Certificates and unresolved instances. Each rejected target stores its query, graph-version hash, rejection category, and a checkable certificate. Component rejection stores a reference witness and its observed atom. Distinctness rejection stores the collapsed witness. Projection rejection stores

the transformation identifier and observed view-row bindings. Non-vacuity rejection stores equal answer-set hashes plus an exact evaluation trace; a non-vacuity pass stores an answer introduced by deletion. All-witness acceptance stores the completed empty rejection-query results and evaluator configuration. These are reproducibility records, not short independently checkable proofs of every negative query result.

A coverage report includes candidate instances, unsupported forms, retained instances, each rejection flag, overlaps, and unresolved cases. Sequential retention counts depend on test order; the overlap report does not. Conditional performance comparisons use prespecified matched strata or weighting and must not compare differently distributed populations without disclosure.

C Formal Properties and Counterexamples

Proposition 1 (All-witness component rejection)**.** *For $a \in A_{\mathrm{ref}}(q)$, membership $a \in U(q)$ holds if and only if at least one reference witness for a uses an observed grounded atom.*

Proof. If $a \in U(q)$, it satisfies some restricted branch query $q^{\mathrm{obs}}_{b,\ell}$. Since $\mathcal{G}_{\mathrm{obs}} \subseteq \mathcal{G}_{\mathrm{ref}}$, the satisfying assignment is a reference witness and its grounded atom ℓ is observed. Conversely, any reference witness containing an observed atom satisfies the corresponding restricted branch query. Taking the union over branches and atoms captures every such witness. No assertion about unrepresented facts outside the finite reference snapshot is used. □

Proposition 2 (Distinctness by diagonal collapse)**.** *Suppose $a \in A_{\mathrm{ref}}(q) \setminus U(q)$, and every statement variable has exactly one main atom. Then $\min_{w \in \mathcal{W}(q,a;\mathcal{G}_{\mathrm{ref}})} d(w) \geq 2$ if and only if $a \notin D(q)$.*

Proof. Every grounded main atom in every reference witness is missing by Proposition 1. Each nonempty branch therefore uses at least one missing record, and $d(w)$ equals the number of distinct records in the image of its statement assignment. This number is one exactly when all statement variables map to the same record. Such an assignment satisfies that branch with all statement variables identified, and conversely every collapsed-branch witness supplies such an assignment. The union over branches gives the result. The proof does not apply before the component audit, where a witness might also use observed records. □

Self-join counterexample. Consider $M(\delta_1, p, r, x) \wedge M(\delta_2, p, r, x) \wedge Q(\delta_1, t, y)$. A single missing record with main triple (p, r, a) and qualifier (t, c) satisfies the query using $\delta_1 = \delta_2$. Counting two variable names would label this as two missing statements incorrectly. The diagonal-collapse test rejects it, even if syntactic duplicate atoms were not removed because their statement variables differ.

Missing edges versus missing statements. A lossless representation of one statement may contain subject, predicate, object, and qualifier edges. Withholding the entire record withholds several such edges but only one hyperfact. Therefore a lower bound on missing representation edges does not imply the same lower bound on distinct missing statements. This is a unit-of-counting distinction, not a claim that binary relations cannot encode qualified facts.

Proposition 3 (Lossless reification)**.** *Let ρ give each record one node with disjointly typed subject, main-relation, object, and qualifier edges. Translate every query statement variable to one shared record-node variable and preserve every entity equality. Then q and $\rho(q)$ have the same entity answers, with corresponding statement witnesses.*

Proof. Map a statement assignment to its record node, preserving entity assignments. Each main and qualifier atom is then satisfied by the corresponding typed edge pattern. Conversely, satisfaction of the translated pattern uses the same record node for every atom of that statement variable; its unique main triple and qualifier edges yield a valid source statement. Repeated qualifier labels are allowed as separate labeled edges. The mapping fails if record nodes are forgotten, qualifier edges are attached to subjects instead, or main and qualifier predicates use an ambiguous shared namespace. □

Projection-splicing counterexample. Let a statement relation have three entity roles A, B, C. Suppose the observed records are (a, b, c'), (a, b', c), and (a', b, c), with primed values distinct from their unprimed counterparts. Its pairwise projections contain (a, b), (a, c), and (b, c), although no observed record contains (a, b, c). Their join produces a spurious complete tuple. A reference-held-out tuple (a, b, c) can consequently be suggested by lower-order projections even when no observed whole statement supports it. This example motivates a projection control; it does not establish its prevalence in any dataset.

Proposition 4 (Exact finite-pool inference)**.** *For finite candidate tables, fixed additive potentials, and a grouped join tree satisfying running intersection, the max-sum recurrence returns the maximum score in the witness score. Its backpointers recover a structurally valid witness whenever that score is finite.*

Proof. Remove a tree edge $i{-}j$. The separator S_{ij} contains every entity variable shared between the two resulting subtrees by running intersection. Conditional on a separator assignment, assignments on the two sides interact only through that separator. A leaf message maximizes its factor score over compatible rows. Inductively, each child message gives the best compatible subtree score, so summing those messages and maximizing the current row gives the best assignment on the entire sending side. Root aggregation therefore yields the branch optimum for each answer value. Maximization over branches yields the query optimum. Backtracking restores compatible rows along each separator; running intersection propagates their shared equalities globally. Each row already groups its statement roles, so reconstruction satisfies the structural verifier. □

Complexity boundary. For a fixed tree and hash-indexed separators, a root-directed pass scans each table and looks up the incoming child messages. Its time is $O(\sum_i (1 + \deg(i))|C_i|)$, excluding proposal scoring, feature computation, hashing arity, and output serialization. A straightforward two-direction pass may cost $O(\sum_i (1 + \deg(i)^2)|C_i|)$; prefix/suffix aggregation can avoid repeated sums, but is not assumed in the empirical cost report. Messages contain at most as many distinct separator assignments as sending-table rows. Incremental updates may revisit the whole tree in the worst case. None of these bounds is polynomial in an unrestricted original graph's size if candidate-table construction itself explodes.

Proposition 5 (Monotone scores, not monotone accuracy)**.** *If $C_i \subseteq C_i'$ for every factor and the potentials are fixed, then $S_\theta(a; C, q) \leq S_\theta(a; C', q)$ for every entity a. The entity's rank and correctness need not improve.*

Proof. Every old witness remains feasible in the expanded pool, so its maximum cannot decrease. A competing entity may acquire a higher-scoring witness, changing the ranking in either direction. Thus score monotonicity gives no monotonicity of MRR, factual support, or utility. □

Duplicate-proof invariance. For fixed canonical candidate tables and unchanged scores, duplicating an identical witness or an alpha-equivalent branch does not change a maximum. This is narrower than invariance to every logically equivalent query. Branch normalization, new factorization choices, or noncanonical finite-budget proposals can change scores or pools. Canonicalization precedes scheduling. RQ5 checks canonical representations after variable renaming, atom and qualifier reordering, and duplicate-branch insertion.

Proposition 6 (Budgeted termination)**.** *Suppose each launched request consumes at least one accounting unit and cannot exceed the remaining finite budget B. Then acquisition launches at most $\lfloor B \rfloor$ requests and terminates. This includes failed and duplicate-only requests.*

Proof. The remaining budget decreases by at least one after each request and stays nonnegative. Thus the number of requests is bounded by the initial budget. An unaffordable indivisible model call is not launched. Exhausted streams or an empty legal action set can cause earlier termination. □

Proposition 7 (Conditional policy improvement)**.** *Fix the proposer, scorer, legal-action generator, and complete state transition. Let $V_t(h)$ be expected terminal utility under policy π_t, and let $Q_t(h, \alpha)$ be expected terminal utility after action α followed by π_t, using the full remaining budget. Suppose the estimate g obeys $\sup_{h,\alpha} |g(h, \alpha) - (Q_t(h, \alpha) - V_t(h))| \leq \epsilon$ on every reachable legal state–action pair. A policy greedy in g, subject to the same mandatory wildcard steps, satisfies*

$V_{t+1}(h) \geq V_t(h) - 2\epsilon\lfloor B(h)\rfloor$, *where* $B(h)$ *is the remaining budget in units with minimum request cost one. In particular, exact advantages imply nondecreasing expected utility.*

Proof. Subtracting $V_t(h)$ does not change action ordering. Uniform error ϵ makes the greedy action's true Q_t at most 2ϵ below the largest legal value. That largest value is at least $V_t(h)$, including when the old policy randomizes over legal actions. Induct on the maximum number of remaining requests. After one request at least one budget unit is consumed, so the induction error from its successor is at most $2\epsilon(\lfloor B(h)\rfloor - 1)$. Adding the one-step comparison gives the bound. A mandatory wildcard has a singleton legal action set and obeys the same inequality. Terminal states have zero error. □

This is an application of finite-horizon policy improvement, not a new general theorem. The error condition requires full-state, full-horizon values on reachable actions. Mean squared training error, sampled rollouts, and a finite feature vector do not establish it. The auxiliary head likewise supplies no such guarantee.

Limits of the formal claims. The propositions do not establish proposal completeness, learned-policy optimality, the absence of all shortcuts, calibrated factual probabilities, or real-world truth. They also do not prove that the strict suite is sufficiently large, balanced, or representative. Those are empirical requirements.

D Persistent Acquisition and Cost Accounting

Stream state. A stream key is (b, j, J, u), where $J \subseteq X_{b,j}$ is the ordered set of conditioned variables and $u : J \to \mathcal{E}$ its assignment. Wildcards use $J = \varnothing$. A cursor stores the current decoding depth, active beam entries, partially scored expansions, emitted complete rows, and an exhausted flag. The retained beam is fixed by canonical ordering, beam width, and deterministic tie-breaking. Suspending a request preserves all these objects. Thus splitting one request into several calls does not change its decoding order.

Entity extensions use the training vocabulary and query constants. The inference vocabulary is declared before evaluation. No hidden reference identifier or answer label enters a stream. A model call is indivisible for budgeting: if a vocabulary-wide call does not fit, that request is unavailable under the remaining allowance. The implementation may instead use a documented index with separately charged retrieval and scoring costs, shared by all matched-generator methods. Candidate scores are frozen during a query.

Recursive resolver. For a deficit $(i \to j, u)$, enqueue the nonexhausted conditioned stream for j. For every acquired row t_j matching u, inspect each neighbor $k \neq i$. If $m_{k\to j}(t_j|_{S_{jk}}) = -\infty$, recursively visit $(j \to k, t_j|_{S_{jk}})$. Deduplicate visited edge–binding keys within one resolver call. Tree traversal excludes the parent and has bounded depth. A cap on visited keys limits overhead; unvisited obligations remain unresolved and can be reconsidered after another acquisition. A deficit can therefore yield requests at both the original factor and downstream factors.

Request exhaustion means the fixed decoding stream has no remaining beam states, not that the underlying graph has no compatible statement. Empty calls whose cursor advances remain eligible; calls that neither advance nor return rows are marked exhausted after a charged attempt. Wildcards use persistent cursors too. The action set includes all affordable nonexhausted wildcards even when both sides of every useful binding are absent. Finite decoding beams can omit true rows permanently, so wildcard scheduling does not establish completeness.

Accounting. Every scored entity extension and complete statement is logged, including discarded and duplicate candidates. Full-vocabulary calls log their entire score vector, batch shape, and device time. Controller features, recursive traversal, and message updates enter latency and memory reports. Weighted scoring work is valid only within a common implementation; across backbones the primary efficiency comparison uses measured end-to-end latency and memory. Fixed streams and frozen scores ensure reproducible transitions from the complete state, including request history and exploration phase.

Table 5: Reference-free inference. All charged operations are logged.

Step	Operation
1	Canonicalize the Core query, group statement roles, and construct a deterministic join tree for each branch.
2	Initialize persistent wildcard streams using a shared, explicitly charged allocation.
3	Compute messages, answer scores, and current witness backpointers.
4	Build the bounded recursive request set, append affordable wildcards, and remove exhausted streams.
5	At a scheduled exploration step select the next affordable wildcard in the persistent round-robin order; otherwise select the highest predicted-utility request.
6	Advance its cursor, charge actual work, insert new unique rows, and recompute messages. Update the budget and exploration phase.
7	Repeat while an affordable request exists. Return ranked entities and witnesses; abstain if every answer score is nonfinite.

E JOINT SUPERVISION AND POLICY ITERATION

Supported maxima. Give each candidate row support bit $v_i(t_i)$, determined by same-record replay against the training reference. Attach binary state z to a message, indicating whether at least one row in its subtree is unsupported:

$$m^{z}_{i\to j}(u) = \max_{\substack{t_i|_{S_{ij}}=u,\ t_i\in C_i \\ z=(1-v_i(t_i))\vee\bigvee_k z_k}} \left[\phi_i(t_i,q) + \sum_{k\in\mathcal{N}(i)\setminus\{j\}} m^{z_k}_{k\to i}(t_i|_{S_{ki}})\right]. \tag{7}$$

Incremental two-state OR convolution combines child messages without enumerating all state combinations. State zero at an answer root gives P_a; state one gives N_a. Ordinary messages over entities outside the complete training answer set give R^-. Empty negative sets contribute zero by definition. This local support decomposition is valid because every occurrence of a statement variable belongs to one grouped factor and Core imposes no inequality across statement variables.

Pool construction and injection. Scorer-training pools combine natural proposer outputs with supported rows needed to supply at least one positive witness for every sampled training answer. Injection is recorded and never applied during controller rollout, validation, or test inference. Report the fraction of training answers requiring injection and compare natural-pool and expanded-pool objectives. A reference-incorrect entity and a correct entity with an unsupported witness are distinct negative categories. Alternative supported witnesses for other correct answers never become negative solely because a different target was sampled.

The proposer is first trained on observed records. Structured training updates the scorer, retaining a local completion objective when its validated coefficient is nonzero. If parameters are shared, all shared parameters are frozen before collecting controller data. The loss does not eliminate false negatives caused by incomplete supervision; the support-masking experiment quantifies this dependence. Scorer-training queries, controller-training queries, and validation queries are disjoint.

Iterative label collection. Initialize π_0 with deficit selection and scheduled wildcards. Collect training states from a fixed mixture of the current policy and uniform legal-action exploration. Sample initial budgets from the full deployment range. Save the complete state, including stream cursors and wildcard phase. At iteration t, freeze π_t and score candidate actions by full-remaining-budget paired continuations. At a mandatory wildcard step the legal action set contains only the scheduled affordable wildcard, so labels cannot train an action that deployment would override. The forced action advances both consumed work and step count. A subsequent wildcard is scheduled from that updated phase, not from a reset counter.

Relabel retained states under π_t before fitting the next regressor; old labels under an earlier policy are not silently mixed into the same target. The heads g_ψ and d_ψ share a two-hidden-layer multilayer perceptron with 64 units per layer and ReLU activations. Inputs are structural features: factor arity, separator dimension, directional support, acquired-row counts, stream progress, request allowance,

remaining budget, and exploration phase. Counts are normalized by the initial query budget. There are no entity IDs, test support bits, or hidden-graph features.

The auxiliary head predicts supported-new-row fraction using training labels only. Its loss is separately weighted, and its prediction is not substituted for terminal utility during selection. Select the number of iterations, mixture weight, and loss weights on validation queries with a fixed search budget. Include the initial heuristic among selectable checkpoints. A validation-selected iterate has no guaranteed advantage on test data.

Limits and cost. Equal rollout allowances make the labels counterfactual comparisons under one fixed continuation. They are not optimal action values. Function approximation, limited action sampling, sparse terminal utility, and visited-state shift remain potential failure sources. With n states, A actions per state, and remaining allowance H, one label iteration costs $O(nAH)$ scoring work before reuse. Report each iteration's actual work, duration, and validation gain.

F Evaluation Metrics and Aggregation

For strict target (q, a), remove all other reference-correct answers to form $E_{q,a} = \mathcal{E} \backslash (A_{\mathrm{ref}}(q) \backslash \{a\})$. The pessimistic filtered rank is $r(q, a) = 1 + \sum_{e \in E_{q,a} \backslash \{a\}} \mathbf{1}[S(e, q) \geq S(a, q)]$. An entity without a complete witness has score $-\infty$. This prevents arbitrary tie-breaking from rewarding an ungenerated target. Degenerate single-entity filtered universes are counted separately.

For the n evaluated targets, let v_i indicate support of the selected witness and c_i indicate existence of any supported witness in the pool. Define

$$\begin{aligned} \mathrm{MRR} &= \frac{1}{n} \sum_i r_i^{-1}, & \mathrm{H@1} &= \frac{1}{n} \sum_i \mathbf{1}[r_i = 1], \\ \mathrm{J@1} &= \frac{1}{n} \sum_i \mathbf{1}[r_i = 1] v_i, & \mathrm{Coverage} &= \frac{1}{n} \sum_i c_i. \end{aligned} \tag{8}$$

Conditional witness support is J@1/H@1, undefined when the denominator is zero. Required checks include $0 \leq \mathrm{J@1} \leq \mathrm{H@1} \leq \mathrm{MRR} \leq 1$ and $\mathrm{J@1} \leq \mathrm{Coverage}$. These dependent outcomes are not counted as independent successes.

The main comparison uses ten paired random seeds and reports the arithmetic mean and sample standard deviation of each seed's target-level metric. Targets sharing a query are dependent observations; they must not be treated as independent replicates for statistical testing. The aggregate results reported here do not determine paired confidence intervals or paired-test probabilities. Accordingly, the main text reports observed differences without significance claims.

Native answer-only models retain their own ranking scores; the missing-witness score rule above applies to witness-based decoders, not to the native answer head of an adapted baseline. Extracted evidence determines only its joint-recovery and coverage metrics. An unsupported or absent selected witness has support indicator zero.

G Baseline Adaptation

Native answers and extracted witnesses. NQE and StarQE retain their official answer heads and training objectives. Their native answer track covers only verified query forms. The evidence track attaches the same declared extractor to each method and preserves the native ranking. Extractor costs and unsupported outputs are logged. The adapter is named explicitly in tables; it is not presented as a native model capability.

Shared scoring and search. The acquisition comparison holds the proposer and statement scorer backbones fixed, including their qualifier representation and training access. Static lists, the deficit heuristic, beam search, best-first search, and HyWit differ in how they allocate candidate requests. Search baselines enforce shared bindings and statement ownership. An inconsistent local chooser is only a failure diagnostic. Pool-replay comparisons give all decoders the union of logged acquired rows and compare their output against exact joins; this isolates decoding from acquisition.

CQD and QTO on reified graphs preserve one record variable per statement, all entity equalities, and separate main/qualifier predicates. Record nodes and relations in the observed representation cannot reveal held-out record identities. Conversion fidelity and supported query coverage are tested before reporting scores. A restricted adapted baseline does not stand in for an unrestricted original benchmark result.

Benchmark controls. Observed exact execution, main-triple lookup, frozen projection lookups, relation frequency, answer frequency, and query relaxation receive the same observation. Projected lookups excluded by the benchmark must fail on retained targets by construction. Residual shortcut evaluation uses held-out transformations and learned statistics. Preprocessing, graph access, validation trials, parameters, and timeouts are separately disclosed.

H Experimental Protocol Details

H.1 Shared Audited Population

The WD50K-NFOL study starts from 4,850 queries and audits eligibility at the query–answer level. Sequential filtration applies the component test, distinctness test, projection controls, and qualifier non-vacuity in that order. Consequently, a rejection count is conditional on passing all earlier stages; summing independent rejection rates would describe a different protocol. All main-result methods are evaluated on the final strict target population.

Distinctness strata. The syntax-forced stratum contains targets whose branch constraints exclude every one-record diagonal collapse. These targets need no graph execution for the distinctness decision. The self-join-compatible stratum instead requires evaluating $D(q)$ in the reference snapshot. Syntax compatibility does not imply that a diagonal witness exists: rejection occurs only when the answer belongs to $D(q)$. For a query with several disjuncts, automatic acceptance requires that every branch preclude the collapse. Otherwise, the compatible branches must be checked.

H.2 Ablation and Transfer Protocols

The evidence-loss ablation has two settings. The fixed-pool setting retains acquired candidates and isolates the scorer change; it cannot change coverage by construction. The end-to-end setting retrains the changed pipeline and can alter acquisition, coverage, and ranking together. These settings answer different questions and are not interchangeable estimates of the same effect.

Structural transfer compares three-factor training-distribution chains with held-out four-factor chains, branching trees, and multi-variable separators. A fixed total budget holds the query allowance constant. A fixed per-factor budget increases total allowance with factor count. Comparisons between the four-factor families control length, while comparisons to the three-factor family include a length change. The results do not establish transfer to cyclic scopes or negation outside Core.

H.3 Semantic Intervention and Support Masking

The qualifier intervention modifies the input query and recomputes its correct answers and supported witnesses against the unchanged reference graph. This avoids scoring a changed query against stale labels. It does not preserve answer prevalence or witness abundance, so performance changes cannot be attributed solely to a model's use of qualifier semantics.

Support masking removes 10%, 20%, or 40% of training support. The evaluation reference is unchanged, allowing coverage and joint recovery to be compared without simultaneously weakening their support criterion. Missing training support can still introduce false negatives; robustness here means relative performance under the specified masking levels, not immunity to incomplete labels.

H.4 Execution Cost

Request allowances are normalized to the base validation allowance and evaluated at 0.25, 0.50, 0.65, 1, 2, and 4 times that budget. Median end-to-end latency is a separate measurement. Equal

abstract request counts do not imply equal neural scoring work or wall-clock time. The quality-matched comparison uses recorded operating points rather than an interpolated crossing.

Wasted-request rates complement recovery but do not measure latency directly: an unsuccessful request can have a different cost from a successful one. An offline amortization analysis additionally requires the incremental training and label-generation cost measured in compatible units. The reported online timings alone do not identify that cost.

I LIMITS AND FALSIFICATION

The exact-inference guarantee assumes a finite candidate pool, fixed additive potentials, and running intersection. It covers neither unrestricted cyclic queries nor proposal completeness. Recursive requests and wildcards can still spend budget on unhelpful candidates. Joint supervision may be harmed by missing reference support, and policy iteration may fail because terminal gains remain sparse or the learned features omit relevant state information.

Benchmark acceptance is relative to a reference snapshot and frozen reduction grammar. It does not establish the absence of statistical shortcuts or reveal a model's internal reasoning process. Whole-record splitting can change the target population; matched controls and marginal statistics are required. A benchmark contribution is weakened if added conditions rarely change eligibility or model assessment.

The algorithm claim is unsupported if strong search controls match its joint recovery at lower total cost, if evidence supervision adds no benefit after equal tuning, or if gains disappear at fixed total budget. The reported comparisons support recovery gains on the evaluated strict targets, not unrestricted query coverage or universal speedups. Offline amortization and paired statistical significance require measurements beyond the aggregate scores reported here.