

QORH: Learning Query-Specific Ordered Rule Hypergraphs for Inductive Knowledge Graph Link Prediction

Naixing Xu^[0009–0005–2094–330X]

School of Software, Shandong University, Jinan, China
naixingxu@mail.sdu.edu.com

Abstract. Knowledge graph link prediction involving unseen entities and relations requires learning transferable reasoning patterns from observed graph structure. Existing methods based on relation graphs primarily capture pairwise dependencies, limiting their ability to preserve the compositional semantics, relation ordering, and role information of complete path rules. To address this limitation, we propose QORH (Query-specific Ordered Rule Hypergraph), which jointly encodes rule heads and ordered rule bodies as hyperedges conditioned on the query relation, explicitly modeling joint dependencies among multiple relations. QORH performs learnable rule selection using statistical evidence, such as rule confidence and coverage, and enhances relation representations through role-aware hypergraph message passing to guide entity-level reasoning. The model’s shared parameters are independent of specific entity or relation identifiers, enabling inductive prediction for unseen entities and relations. Extensive experiments on entity-inductive and fully inductive benchmarks demonstrate that QORH significantly outperforms the state-of-the-art baselines included in our evaluation, validating the effectiveness of rule hypergraph modeling for inductive link prediction.

Keywords: Knowledge graph link prediction · Inductive reasoning · Ordered rule hypergraphs · Higher-order relational modeling · Graph neural networks.

1 Introduction

Knowledge graphs (KGs) represent entities and their semantic relationships as structured triples (h, r, t) , where h , t , and r denote the head entity, tail entity, and their relation, respectively. Owing to their rich semantics and structured representation capabilities, KGs have been widely applied to recommender systems [1, 2], question answering [3, 4], biomedicine [5, 6], and other domains [7, 8].

However, KGs are typically incomplete because of the high cost of knowledge acquisition and the continuous evolution of real-world knowledge. Knowledge graph link prediction aims to infer missing facts from observed ones and has become a central task in KG research. Although conventional embedding methods

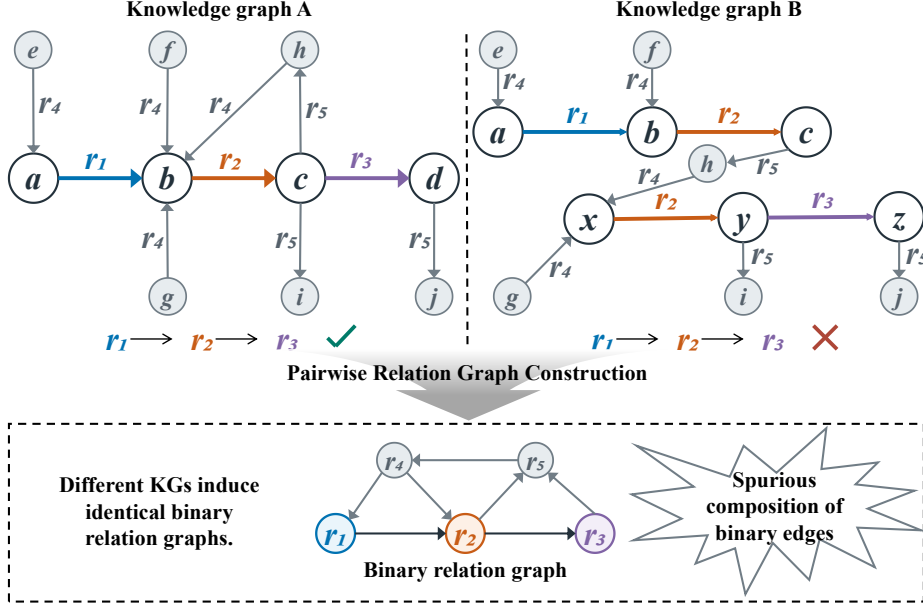


Fig. 1. Spurious composition of binary edges in relation-level auxiliary graphs. The two KGs induce identical binary relation graphs, yet only KG A contains a continuous path with the relation sequence (r_1, r_2, r_3) .

such as TransE [9], DistMult [10], ComplEx [11], and RotatE [12] have achieved notable progress, they generally follow the transductive setting, assuming that all entities and relations encountered during inference have been observed during training. Consequently, they struggle to generalize to unseen entities and relations.

To handle unseen entities, entity-inductive methods such as GraIL [13], TACT [14], MorsE [15], and Content Pooling [16] learn transferable local structures and relational patterns without relying on entity-specific representations. Nevertheless, these methods typically assume a shared relation set between the training and test graphs, limiting their applicability to open-world scenarios in which both entities and relations continuously emerge.

Accordingly, fully inductive KG link prediction requires models to generalize simultaneously to unseen entities and relations. Methods such as RMPI [17], InGram [18], and TARGI [19] enable fully inductive reasoning by learning transferable relational structures. KG foundation models such as ULTRA [20] and GraphOracle [21] further learn general reasoning patterns independent of specific entity and relation vocabularies through pretraining across multiple KGs, achieving strong performance on various inductive benchmarks with limited fine-tuning on target KGs.

Despite the progress made by fully inductive KG link prediction methods such as InGram [18], ULTRA [20], TARGI [19], and GraphOracle [21], their relation

modeling is generally limited to pairwise dependencies induced by shared entities, making it difficult to preserve the compositional semantics, relation roles, and overall reliability of ordered multi-relation rules. As illustrated in Figure 1, given a KG $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{F})$, GraphOracle [21] establishes a directed edge in its relation-level auxiliary graph whenever two relations are sequentially connected through an intermediate entity: $(r_i, r_j) \in \mathcal{E}_R \iff \exists u, x, v \in \mathcal{V} : (u, r_i, x) \in \mathcal{F} \wedge (x, r_j, v) \in \mathcal{F}$, where $\mathcal{V}$, $\mathcal{R}$, and $\mathcal{F}$ denote the entity set, relation set, and factual triple set, respectively, and $\mathcal{E}_R$ is the directed edge set of the relation-level auxiliary graph. This construction retains only local adjacency between relations and cannot determine whether multiple pairwise dependencies originate from the same complete path. For example, both knowledge graph A and knowledge graph B contain $r_1 \rightarrow r_2$ and $r_2 \rightarrow r_3$, whereas only knowledge graph A contains the complete sequence $r_1 \rightarrow r_2 \rightarrow r_3$. In knowledge graph B, the two relation pairs arise from different paths and therefore cannot form a valid reasoning chain. We refer to this limitation as the “spurious composition of binary edges,” indicating that pairwise dependencies alone are insufficient to recover higher-order joint structures among multiple relations.

Moreover, as discussed in GraphOracle, the performance of existing KG foundation models often depends substantially on fine-tuning over target datasets [21]. Our data analysis in Section 4.2 further reveals overlaps between some pretraining data and inductive test sets, exposing models to information that should remain unseen during evaluation and thereby causing data leakage. Consequently, the resulting performance may conflate pretraining memorization with inductive generalization, making it difficult to assess the models’ true capabilities under strict inductive protocols. Therefore, achieving reliable link prediction for unseen entities and even unseen relations under strict inductive evaluation protocols remains an important open problem.

To address these limitations, we propose a query-specific rule hypergraph enhancement method that represents ordered path rules as hyperedges connecting rule-head and rule-body relations, thereby preserving their compositional semantics, ordering, and role information. A lightweight multilayer perceptron jointly considers rule statistics, such as confidence and coverage, to select high-quality hyperedges, while role-aware relation–hyperedge–relation message passing enhances relation representations for entity-level reasoning and link prediction. Following the entity-inductive protocol of GraIL [13] and the fully inductive protocol of InGram [18], our method achieves state-of-the-art performance in both inductive settings. The main contributions of this work are summarized as follows:

- We propose a query-specific rule hypergraph enhancement method that explicitly models ordered multi-relation rules through learnable rule selection and role-aware hypergraph propagation.
- We identify the loss of multi-relation rule information and the spurious composition of binary edges caused by binary relation-level auxiliary graphs, and analyze this limitation from the perspective of higher-order relational modeling.

- We conduct systematic experiments on 12 entity-inductive and 12 fully-inductive datasets. The proposed method consistently outperforms existing state-of-the-art approaches on the major evaluation metrics, demonstrating strong inductive generalization and robustness.

2 Related Work

This section systematically reviews recent advances in entity-inductive and fully-inductive knowledge graph link prediction and surveys related work on hyper-graphs for knowledge graph representation learning and link prediction.

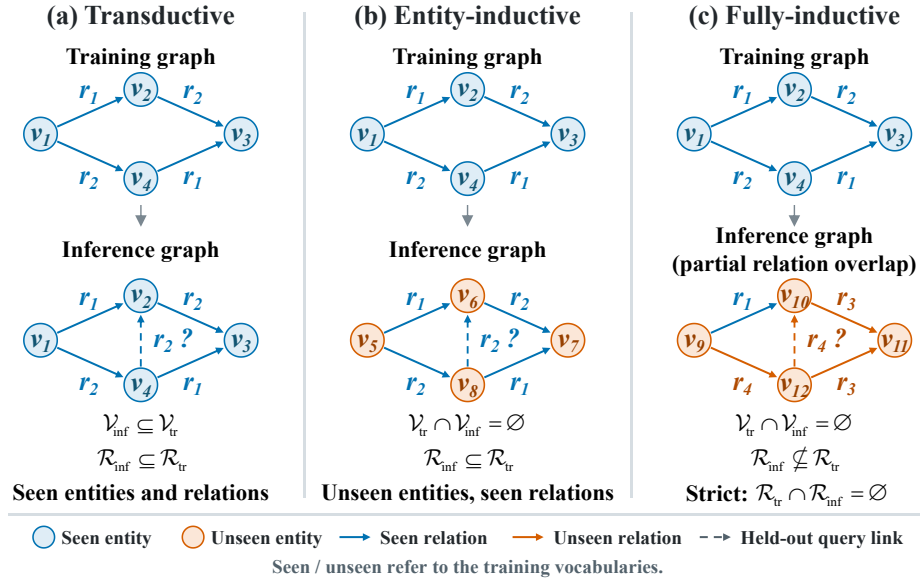


Fig. 2. Illustration of knowledge graph link prediction settings. (a) Transductive: seen entities and relations. (b) Entity-inductive: unseen entities with seen relations. (c) Fully-inductive: unseen entities with partially or entirely unseen relations.

2.1 Entity-Inductive Knowledge Graph Link Prediction

As illustrated in Figure 2, let $\mathcal{V}_{\text{tr}}$ and $\mathcal{V}_{\text{inf}}$ denote the entity sets of the training and inference graphs, respectively, and let $\mathcal{R}_{\text{tr}}$ and $\mathcal{R}_{\text{inf}}$ denote their corresponding relation sets. Under the transductive setting, all entities and relations involved at inference time have already appeared during training; that is, $\mathcal{V}_{\text{inf}} \subseteq \mathcal{V}_{\text{tr}}$ and $\mathcal{R}_{\text{inf}} \subseteq \mathcal{R}_{\text{tr}}$. In contrast, the entity-inductive setting requires $\mathcal{V}_{\text{tr}} \cap \mathcal{V}_{\text{inf}} = \emptyset$ while retaining a shared relation vocabulary, thereby evaluating the model’s ability to generalize to unseen entities.

Logical rules are inherently independent of specific entities and therefore constituted an important early approach to entity-inductive reasoning. Neural LP [22] represents relation compositions as differentiable matrix operations; DRUM [23] improves the efficiency of learning long rules through recurrent rule modeling; and RNNLogic [24] treats logical rules as latent variables and jointly optimizes rule generation and reasoning.

More recently, graph neural networks have further advanced this field. GraIL [13] encodes the local enclosing subgraph surrounding each candidate triple to learn entity-independent structural patterns. CoMPiLE [11] strengthens the interactions between entities and relations within directed subgraphs, thereby improving the modeling of asymmetric relations. TACT [14] captures topological correlations between relation pairs to enrich the relational information contained in local subgraphs. MorsE [15] employs meta-learning to generate representations of unseen entities from their observed neighborhoods. Among path-based reasoning methods, NBFNet [25] formulates link prediction as a learnable generalized Bellman–Ford process; RED-GNN [26] improves reasoning efficiency by sharing overlapping paths among candidate entities; and the recently proposed Context Pooling [16] selects logically relevant neighbors conditioned on a given query to suppress structural noise.

Although these methods can generalize to unseen entities, they typically assume that the training and inference graphs share the same relational semantics, allowing learned relation-specific parameters or rule templates to be reused during inference. Consequently, they have difficulty adapting to open knowledge graphs in which new relations continually emerge.

2.2 Fully-Inductive Knowledge Graph Link Prediction

As illustrated in Figure 2(c), the fully-inductive setting further allows the inference graph to contain relations unseen during training, i.e., $\mathcal{R}_{\text{inf}} \not\subseteq \mathcal{R}_{\text{tr}}$. Under the strictest setting, $\mathcal{R}_{\text{tr}} \cap \mathcal{R}_{\text{inf}} = \emptyset$ also holds. Therefore, the model must derive representations of both unseen entities and unseen relations from the structure of the inference graph.

Existing methods primarily employ relation-level auxiliary graphs to learn transferable representations. RMPI [17] performs relational message passing within local subgraphs; InGram [18] constructs a weighted relation graph based on relation affinities and successively aggregates relation-level and entity-level information; and TARGI [19] exploits global topological interactions to alleviate the biases introduced by local structures and relation frequencies.

In recent years, knowledge graph foundation models have demonstrated strong reasoning performance and generalization capabilities in inductive link prediction. ULTRA [20] learns transferable relative relation representations by categorizing the ways in which relations share entities into four interaction types: head-to-head, head-to-tail, tail-to-head, and tail-to-tail. The recently proposed GraphOracle [21] constructs a sparse directed relation-dependency graph and generates relation representations through query-conditioned attention.

However, the relation-level auxiliary graphs employed by these methods are still predominantly composed of pairwise edges. Although such graphs can represent co-occurrence, topological, or logical dependencies between relations, they cannot reliably determine whether multiple pairwise dependencies originate from the same complete path. Consequently, the overall semantics of a rule, the positions of relations along the path, and their respective roles may be weakened. Furthermore, knowledge graph foundation models such as ULTRA [20] and GraphOracle [21] require lengthy pretraining on multiple datasets, including WN18RR [27] and FB15k-237 [28], whereas existing mainstream inductive link prediction benchmarks [13, 18] are constructed by partitioning WN18RR [27] and FB15k-237 [28]. Therefore, even when inductive evaluation is conducted in a zero-shot setting, i.e., without fine-tuning on the target knowledge graph, overlaps may still exist between the training data and the inductive test data (see Section 4.2). As a result, the reported performance may not faithfully reflect the models’ generalization capabilities under strict inductive evaluation protocols.

Unlike the aforementioned methods, our approach strictly follows the inductive evaluation protocols introduced by GraIL [13] and InGram [18]. For each query relation, it mines complete ordered path rules and represents each rule as a hyperedge connecting the rule head and multiple rule-body relations, thereby explicitly modeling multi-relation joint dependencies that cannot be fully captured by binary relation graphs.

2.3 Hypergraphs for Knowledge Graph Representation and Link Prediction

A hypergraph allows a single hyperedge to connect more than two nodes, enabling it to represent higher-order dependencies that ordinary graphs cannot fully capture. HGNN [29] learns higher-order data correlations through message passing and provides a general framework for subsequent research on hypergraph neural networks. Several studies have extended hypergraph modeling to more complex knowledge representation and knowledge graph link prediction scenarios. StarE [30] develops a message-passing mechanism for hyper-relational knowledge graphs with qualifiers, jointly encoding a primary triple and its auxiliary information. LHP [31] represents logical conjunctions using directed hyperedges and treats logical dependencies among multiple nodes as an integrated structure for hyperlink prediction. ReALE [32] further improves the expressiveness of knowledge hypergraph representations from the perspective of relational algebra. Nevertheless, existing methods primarily focus on representing and completing native n -ary facts or qualified facts, and most operate under transductive settings in which the training and test stages share entity or relation vocabularies. By contrast, the application of hypergraphs to inductive knowledge graph link prediction remains at an early stage. In particular, how to construct transferable relation-level reasoning structures using hypergraphs in traditional knowledge graphs, where binary relations are represented as triples (h, r, t) , remains insufficiently explored.

3 Methodology

This section details the proposed Query-specific Ordered Rule Hypergraph (QORH) method, whose overall framework is illustrated in Figure 3.

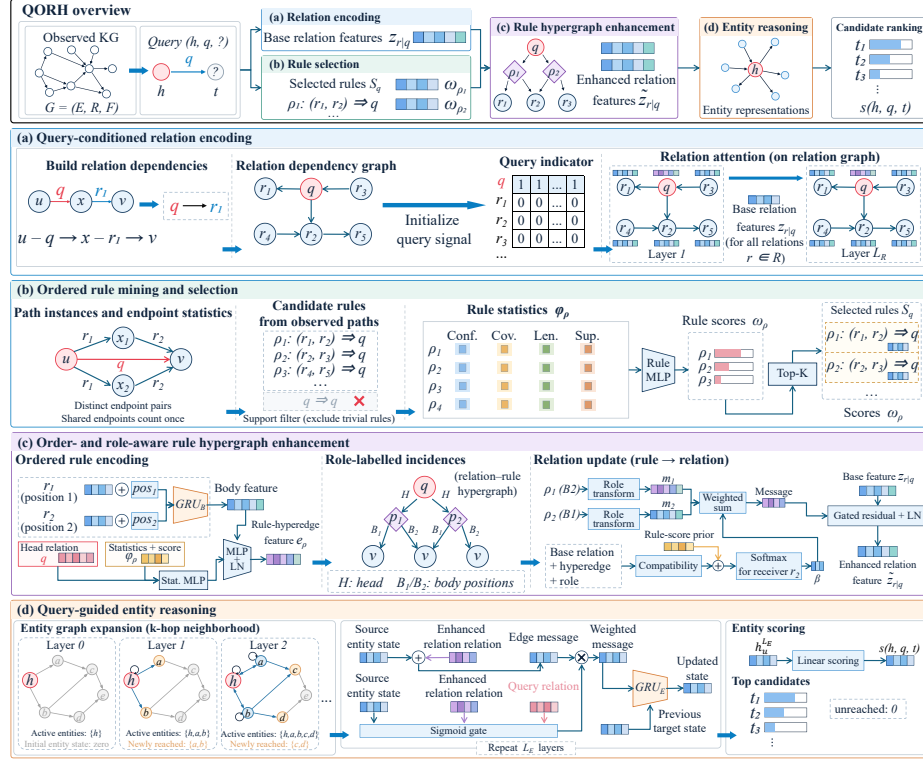


Fig. 3. Overview of QORH: query-specific ordered rule hypergraphs for inductive knowledge graph link prediction.

3.1 Problem Formulation and Framework Overview

Following the entity-inductive and fully inductive settings defined in Section 2, let $\mathcal{G}_{\text{obs}} = (\mathcal{V}, \mathcal{R}, \mathcal{F}_{\text{obs}})$ denote the observed knowledge graph at the current stage, where $\mathcal{V}$, $\mathcal{R}$, and $\mathcal{F}_{\text{obs}}$ are the sets of entities, relations, and observed facts available for reasoning, respectively. Given a head entity h and a query relation q , the model ranks candidate tail entities $t \in \mathcal{V}$ for the query $(h, q, ?)$ using the scoring function $s_\theta(h, q, t | \mathcal{G}_{\text{obs}})$, where θ denotes the learnable parameters.

To unify head and tail entity prediction, we augment each observed fact (u, r, v) with its inverse fact (v, r^{-1}, u) , yielding the fact set $\mathcal{F}_\pm$ and the relation

set $\mathcal{R}_\pm = \mathcal{R} \cup \mathcal{R}^{-1}$, where $\mathcal{R}^{-1}$ is the set of inverse relations. We further add a self-loop (v, r_{self}, v) to each entity, yielding the fact set $\tilde{\mathcal{F}}$ and the relation set $\overline{\mathcal{R}} = \mathcal{R}_\pm \cup r_{\text{self}}$ for entity-level message passing. Rules are mined exclusively from $\mathcal{F}_\pm$, excluding the artificially added entity self-loops. QORH first generates query-conditioned base relation representations $\mathbf{z}_{r|q} \in \mathbb{R}^d$ on the relation dependency graph, then obtains enhanced representations $\tilde{\mathbf{z}}_{r|q}$ through the rule hypergraph, and finally uses these enhanced representations to perform entity-level message passing from the head entity h and score candidate tail entities. Here, d is the hidden dimension, and the subscript $r | q$ indicates that the representation of relation r is conditioned on the query relation q .

3.2 Query-Conditioned Relation Encoding

As shown in Figure 3(a), QORH first extracts local relational context from the observed graph to obtain base representations for the subsequent encoding of complete rules. The base relation encoder follows the relation graph modeling approaches used in prior work [18, 20, 21], treating relation types as nodes and converting local connectivity patterns in the knowledge graph into structural links between relations. Specifically, we construct a directed relation graph based on whether two entity-level edges form a head-to-tail path through an intermediate entity:

$$\mathcal{E}_R^\rightarrow = \{(r, s) \in \mathcal{R}_\pm^2 \mid \exists u, x, v \in \mathcal{V} : (u, r, x) \in \mathcal{F}_\pm \wedge (x, s, v) \in \mathcal{F}_\pm\}. \quad (1)$$

Here, r and s are the relations of the first and second edges along the path, respectively, and x is the intermediate entity connecting the two edges. To enable information exchange between adjacent relations, we augment this dependency set with reverse propagation edges and add a self-loop to each relation node, yielding the relation graph $\mathcal{G}_R = (\overline{\mathcal{R}}, \mathcal{E}_R)$, where

$$\mathcal{E}_R = \mathcal{E}_R^\rightarrow \cup \{(s, r) \mid (r, s) \in \mathcal{E}_R^\rightarrow\} \cup \{(r, r) \mid r \in \overline{\mathcal{R}}\}. \quad (2)$$

These reverse propagation edges connect relation nodes and are distinct from the inverse relations that represent inverse facts in the entity graph. Inspired by the query-conditioned relation dependency graph modeling approach in GraphOracl [21], QORH initializes relation nodes with query indicators for a given query relation q :

$$\mathbf{z}_{r|q}^{(0)} = \begin{cases} \mathbf{1}_d, & r = q, \\ \mathbf{0}_d, & r \neq q, \end{cases} \quad r \in \overline{\mathcal{R}}, \quad (3)$$

where $\mathbf{1}_d$ and $\mathbf{0}_d$ are the d -dimensional all-ones and zero vectors, respectively. This initialization provides an initial signal at the query relation for structural propagation, allowing different query relations to obtain their own contextual representations. Let $\mathcal{N}_R(s) = \{r \mid (r, s) \in \mathcal{E}_R\}$ denote the set of incoming neighbors of relation node s , including s itself. At propagation layer k , the attention

score and normalized weight for edge (r, s) are given by

$$\begin{aligned} e_{r,s|q}^{(k)} &= \left(\mathbf{a}_R^{(k)} \right)^\top \left[\mathbf{W}_A^{(k)} \mathbf{z}_{r|q}^{(k-1)} \parallel \mathbf{W}_A^{(k)} \mathbf{z}_{s|q}^{(k-1)} \right], \\ \alpha_{r,s|q}^{(k)} &= \frac{\exp\left(e_{r,s|q}^{(k)}\right)}{\sum_{u \in \mathcal{N}_R(s)} \exp\left(e_{u,s|q}^{(k)}\right)}. \end{aligned} \quad (4)$$

Here, $\parallel$ denotes vector concatenation, $\mathbf{W}_A^{(k)} \in \mathbb{R}^{d \times d}$ is the attention projection matrix, and $\mathbf{a}_R^{(k)} \in \mathbb{R}^{2d}$ is a learnable attention vector. Normalization is performed over the incoming edges of the receiving relation s for a fixed query q . The relation representation is updated as

$$\mathbf{z}_{s|q}^{(k)} = \text{ReLU} \left(\mathbf{W}_O^{(k)} \left[\mathbf{W}_N^{(k)} \sum_{r \in \mathcal{N}_R(s)} \alpha_{r,s|q}^{(k)} \mathbf{z}_{r|q}^{(k-1)} + \mathbf{W}_S^{(k)} \alpha_{s,s|q}^{(k)} \mathbf{z}_{s|q}^{(k-1)} \right] \right). \quad (5)$$

Here, $\mathbf{W}_N^{(k)}$, $\mathbf{W}_S^{(k)}$, and $\mathbf{W}_O^{(k)}$ are learnable $d \times d$ matrices for neighborhood aggregation, transformation of the node's own information, and output mapping, respectively; ReLU is an element-wise activation function. Neighborhood aggregation includes self-loop messages, while a separate transformation of the node's own information preserves its query-dependent state. After L_R propagation layers, we set $\mathbf{z}_{r|q} = \mathbf{z}_{r|q}^{(L_R)}$, where L_R is the depth of the relation encoder. The resulting representation captures the local structural context of relation r relative to query q and serves as the input to the subsequent rule hypergraph encoder.

3.3 Query-Specific Rule Hypergraph Construction

The relation graph provides QORH with query-dependent local context, but its pairwise edges do not explicitly indicate whether multiple dependencies originate from the same complete path. To address this limitation, QORH further mines ordered path rules from the observed graph and represents each selected rule in its entirety as a hyperedge. Figure 3(b) illustrates the process from rule mining to learnable selection.

Mining Ordered Path Rules To associate complete paths with their conclusion relations, QORH mines ordered path rules of the following form from the observed facts:

$$\rho : r_1(x_0, x_1) \wedge r_2(x_1, x_2) \wedge \cdots \wedge r_{\ell_\rho}(x_{\ell_\rho-1}, x_{\ell_\rho}) \Rightarrow q(x_0, x_{\ell_\rho}). \quad (6)$$

Here, ρ denotes a rule, q is its head relation, $(r_1, \dots, r_{\ell_\rho})$ is its ordered body, and ℓ_ρ is its body length, satisfying $1 \leq \ell_\rho \leq L_{\max}$; $L_{\max}$ is a configurable maximum rule length. The variables $x_0, \dots, x_{\ell_\rho}$ denote entities along the path,

and adjacent relations in the rule body are linked through shared variables. Thus, a rule specifies not only which relations co-occur but also their composition order and the correspondence between their endpoints.

Specifically, we enumerate simple directed paths over $\mathcal{F}_\pm$ that do not revisit any entity and group paths with identical relation sequences into the same rule body pattern. A relation type may occur at multiple positions along a path. To control computational cost, we cap the number of outgoing edges retained per entity and the number of paths enumerated per starting entity at each depth, with default limits of 64 and 1024, respectively. Let $\mathcal{P}$ denote the set of enumerated paths; for a rule ρ , we define the set of endpoint entity pairs associated with its body and the set of head facts as

$$\begin{aligned}\mathcal{B}_\rho &= \{(v_0, v_{\ell_\rho}) \mid (v_0, r_1, v_1, \dots, r_{\ell_\rho}, v_{\ell_\rho}) \in \mathcal{P}\}, \\ \mathcal{H}_q &= \{(u, v) \mid (u, q, v) \in \mathcal{F}_\pm\}.\end{aligned}\quad (7)$$

Here, $v_0, \dots, v_{\ell_\rho}$ are the entities in a concrete path; $\mathcal{B}_\rho$ contains the distinct entity pairs connected by the enumerated rule body paths, and $\mathcal{H}_q$ contains the distinct entity pairs directly connected by relation q . Computing statistics over sets of entity pairs ensures that each endpoint pair is counted only once, even when multiple paths instantiate the same rule body between the two endpoints. The support, confidence, and coverage of a rule are defined as

$$s_\rho = |\mathcal{B}_\rho \cap \mathcal{H}_q|, \quad \text{conf}_\rho = \frac{s_\rho}{|\mathcal{B}_\rho|}, \quad \text{cov}_\rho = \frac{s_\rho}{|\mathcal{H}_q|}. \quad (8)$$

Here, s_ρ counts the endpoint entity pairs that satisfy both the rule body and the rule head, quantifying the observed support for the rule; conf_ρ measures the proportion of entity pairs connected by the rule body that also have the corresponding head fact; cov_ρ measures the proportion of observed head facts that can be explained by the rule body. These statistics characterize the amount of supporting evidence, conditional reliability, and coverage of observed facts, respectively. We retain only candidate rules whose support is at least the threshold $s_{\min}$ and exclude tautological identity rules of the form $q(x, y) \Rightarrow q(x, y)$. Let $\mathcal{C}_q$ denote the resulting set of candidate rules with head relation q .

For example, support for $r_1(x, z) \wedge r_2(z, y) \Rightarrow q(x, y)$ requires observing both the complete two-hop path and relation q for the same endpoint pair (x, y) . This unit of statistical analysis jointly retains the complete path and its conclusion relation, enabling subsequent modeling to exploit joint information that is absent from independent relation pairs.

Learnable Rule Selection The mining procedure yields candidate rules with observed support, but these rules still vary in the amount of support, coverage, and path length. To align rule use with the link prediction objective, QORH introduces a learnable selector before hyperedge encoding. Specifically, we construct a four-dimensional feature vector for each candidate rule:

$$\phi_\rho = \left[\frac{s_\rho + 1}{|\mathcal{B}_\rho| + 2}, \frac{s_\rho + 1}{|\mathcal{H}_q| + 2}, \frac{\ell_\rho}{L_{\max}}, \frac{\log(1 + s_\rho)}{\log(1 + s_{\max})} \right]^\top \in \mathbb{R}^4. \quad (9)$$

Here, $s_{\max}$ is the maximum support among all candidate rules in the current observed graph, taken over the union of all nonempty candidate sets $\mathcal{C}_q$. The first two components are the confidence and coverage with add-one smoothing, respectively, which mitigates extreme estimates arising from limited observations; the last two components normalize rule length and support, respectively, with the logarithmic transformation reducing numerical differences across support scales.

The rule selection score is computed by a shared multilayer perceptron (MLP):

$$\omega_\rho = \sigma(\mathbf{w}_{s2}^\top \text{SiLU}(\mathbf{W}_{s1}\phi_\rho + \mathbf{b}_{s1}) + b_{s2}). \quad (10)$$

Here, $\mathbf{W}_{s1} \in \mathbb{R}^{d_s \times 4}$, $\mathbf{b}_{s1} \in \mathbb{R}^{d_s}$, $\mathbf{w}_{s2} \in \mathbb{R}^{d_s}$, and $b_{s2} \in \mathbb{R}$ are learnable parameters, and d_s is the hidden dimension of the selector; σ is the sigmoid function, and $\text{SiLU}(x) = x\sigma(x)$ is the hidden-layer activation function; $\omega_\rho \in (0, 1)$ denotes the rule selection score.

For query relation q , we select up to K rules with the highest scores from $\mathcal{C}_q$:

$$k_q = \min(K, |\mathcal{C}_q|), \quad \mathcal{S}_q = \text{TopK}_{k_q}(\mathcal{C}_q; \omega_\rho). \quad (11)$$

Here, K is a configurable upper bound on the number of selected rules, k_q is the actual number selected, and $\mathcal{S}_q$ is the selected rule set; TopK selects rules in descending order of ω_ρ . If $\mathcal{C}_q$ is empty, we set $\mathcal{S}_q = \emptyset$.

Rule selection precedes hyperedge encoding, restricting the more computationally expensive sequence encoding and hypergraph propagation to the selected rules. The selection score ω_ρ also contributes to hyperedge feature encoding and message attention, linking rule filtering and the use of rule information through the same score.

Order- and Role-Preserving Hyperedge Representation After selecting the rules, QORH converts these discrete rules into hyperedge representations suitable for neural message passing. For each $\rho \in \mathcal{S}_q$, we construct a hyperedge connecting the rule head and body relations. Recording only the relations contained in a hyperedge is insufficient to distinguish different ordered rules, so we also preserve each relation’s position and role within the rule.

We define

$$r(\rho, j) = \begin{cases} q, & j = 0, \\ r_j, & 1 \leq j \leq \ell_\rho, \end{cases} \quad \mathcal{I}_q = \{(r(\rho, j), \rho, j) \mid \rho \in \mathcal{S}_q, 0 \leq j \leq \ell_\rho\}. \quad (12)$$

Here, $j = 0$ denotes the rule head role, while $j \geq 1$ denotes position j in the rule body; $r(\rho, j)$ returns the relation node associated with that role; $\mathcal{I}_q$ is the set of role-labeled relation-hyperedge incidences.

The rule hypergraph for the query relation is denoted by $\mathcal{G}_{H,q} = (\overline{\mathcal{R}}, \mathcal{S}_q, \mathcal{I}_q)$, with relation types as nodes and a one-to-one correspondence between hyperedges and selected rules. Even when the same relation occurs at multiple positions in a rule body, the incidence set retains each position separately, preserving the distinctions between roles within the rule. Figure 3(c) illustrates the

construction of hyperedge representations, in which the rule body branch first encodes the ordered composition of relations. For the relation at body position j of rule ρ , we construct a position-dependent input and sequentially update the hidden state using a gated recurrent unit (GRU):

$$\begin{aligned} \mathbf{x}_{\rho,j} &= \mathbf{W}_B \mathbf{z}_{r_j|q} + \mathbf{p}_j, \\ \mathbf{b}_{\rho,j} &= \text{GRU}_B(\mathbf{x}_{\rho,j}, \mathbf{b}_{\rho,j-1}), \quad \mathbf{b}_{\rho,0} = \mathbf{0}_d, \\ \mathbf{b}_\rho &= \mathbf{b}_{\rho,\ell_\rho}. \end{aligned} \quad (13)$$

Here, $\mathbf{W}_B \in \mathbb{R}^{d \times d}$ is a projection matrix, $\mathbf{p}_j \in \mathbb{R}^d$ is the learnable embedding for body position j , and $\mathbf{x}_{\rho,j} \in \mathbb{R}^d$ is the input feature at that position; GRU_B is a sequence encoder shared across all rules, $\mathbf{b}_{\rho,j}$ is its hidden state after processing the first j relations, and $\mathbf{b}_\rho$ is the representation of the complete rule body. Encoding is applied only to valid positions within each rule. Positional embeddings identify each relation’s position along the path, while recurrent updates compose relation information in path order.

We then encode the rule statistics and selection score into a statistical representation and fuse it with the head and body representations:

$$\begin{aligned} \mathbf{c}_\rho &= \text{MLP}_{\text{stat}}([\phi_\rho \parallel \omega_\rho]), \\ \mathbf{e}_\rho &= \text{LN}_e(\text{MLP}_{\text{edge}}([\mathbf{z}_{q|q} \parallel \mathbf{b}_\rho \parallel \mathbf{c}_\rho])). \end{aligned} \quad (14)$$

Here, $\text{MLP}_{\text{stat}} : \mathbb{R}^5 \rightarrow \mathbb{R}^d$ maps the four statistical features and the selection score to $\mathbf{c}_\rho \in \mathbb{R}^d$; $\text{MLP}_{\text{edge}} : \mathbb{R}^{3d} \rightarrow \mathbb{R}^d$ fuses the three types of input, with both MLPs implemented as two-layer perceptrons with GELU activation and hidden dimension d ; LN_e denotes layer normalization; $\mathbf{e}_\rho \in \mathbb{R}^d$ is the resulting hyperedge representation. The three components provide the query relation targeted by the rule, the ordered composition pattern of its body, and the statistical evidence for the rule, respectively. QORH thus encodes each rule into a unified representation incorporating its head, body, and statistical evidence. For example, the representation of rule $r_1(x, z) \wedge r_2(z, y) \Rightarrow q(x, y)$ preserves the role of q as the conclusion, the relative positions of r_1 and r_2 , and the statistics of their joint support.

The hyperedge representation specifies how a complete rule is encoded; the next step is to determine how this information contributes to relation nodes serving different roles.

3.4 Role-Aware Hypergraph Message Passing

To propagate complete rule information back to the relation level, QORH performs hyperedge-to-relation message passing along role-labeled incidences, which, together with the relation-to-hyperedge encoding in the preceding section, constitutes the relation enhancement process. As shown in Figure 3(d), this process first generates messages according to rule roles, then aggregates them using incidence compatibility and rule selection scores, and finally updates relation representations through gated residual connections.

Role-Specific Messages and Rule-Quality Attention The head and body relations of a rule serve different functions in the reasoning structure, and different positions within the body correspond to different stages of relation composition. QORH therefore uses role-specific transformations to map the same hyperedge representation to messages for different participating relations. For a relation-hyperedge incidence $(r(\rho, j), \rho, j)$, we define

$$\boldsymbol{\mu}_{\rho,j} = \mathbf{W}_j^{\text{msg}} \mathbf{e}_\rho, \quad \mathbf{W}_j^{\text{msg}} = \begin{cases} \mathbf{W}_{\text{head}}, & j = 0, \\ \mathbf{W}_{\text{body},j}, & 1 \leq j \leq L_{\max}. \end{cases} \quad (15)$$

Here, $\boldsymbol{\mu}_{\rho,j} \in \mathbb{R}^d$ is the message sent by rule ρ to the relation in role j ; $\mathbf{W}_{\text{head}} \in \mathbb{R}^{d \times d}$ is the message transformation for the rule head, and $\mathbf{W}_{\text{body},j} \in \mathbb{R}^{d \times d}$ is the message transformation for body position j . A relation may participate in multiple rules, so its incoming messages should account for rule content, the state of the receiving relation, and rule quality. To this end, we combine relation-hyperedge incidence compatibility with the selection score to compute attention:

$$\xi_{\rho,j|q} = \mathbf{w}_H^\top \tanh(\mathbf{W}_{Hr} \mathbf{z}_{r(\rho,j)|q} + \mathbf{W}_{He} \mathbf{e}_\rho + \mathbf{u}_j) + \eta \log(\max(\omega_\rho, \varepsilon)). \quad (16)$$

Here, $\xi_{\rho,j|q}$ denotes the unnormalized attention score, conditioned on query relation q , for the relation-hyperedge incidence indexed by (ρ, j) . The matrices $\mathbf{W}_{Hr}, \mathbf{W}_{He} \in \mathbb{R}^{d \times d}$ map the representation of the receiving relation node and the rule hyperedge representation, respectively, into a shared attention feature space. The symbol ρ denotes the rule associated with the incidence, and j denotes the role index, with $j = 0$ corresponding to the rule head and $j \geq 1$ to the j -th position in the rule body. The notation $r(\rho, j)$ identifies the relation node associated with that role, and $\mathbf{z}_{r(\rho,j)|q} \in \mathbb{R}^d$ denotes its base representation conditioned on query relation q , before hypergraph enhancement. The vector $\mathbf{e}_\rho \in \mathbb{R}^d$ represents the hyperedge corresponding to rule ρ , while $\mathbf{u}_j \in \mathbb{R}^d$ is a role embedding used to distinguish a relation's position and function within the rule when computing compatibility. The vector $\mathbf{w}_H \in \mathbb{R}^d$ is the attention scoring vector, and $\tanh$ denotes the element-wise hyperbolic tangent function. The rule-selection score $\omega_\rho \in (0, 1)$ is produced by the rule selector, $\eta \geq 0$ controls its strength as an attention prior, and $\varepsilon = 10^{-8}$ ensures numerical stability in the logarithm computation. This logarithmic prior allows rule-selection scores to further modulate the relative contributions of incidence messages beyond content compatibility.

For a given query relation q and receiving relation node r , let $\mathcal{I}_q(r) = \{(\rho, j) \mid (r, \rho, j) \in \mathcal{I}_q\}$ denote the set of rule-role pairs incident to that node. If $\mathcal{I}_q(r)$ is nonempty, we apply softmax normalization to the attention scores of these incidences and aggregate the corresponding role-specific messages using the resulting

weights:

$$\begin{aligned}\beta_{\rho,j|q} &= \frac{\exp(\xi_{\rho,j|q})}{\sum_{(\rho',j') \in \mathcal{I}_q(r)} \exp(\xi_{\rho',j'|q})}, & (\rho,j) \in \mathcal{I}_q(r), \\ \mathbf{m}_{r|q} &= \sum_{(\rho,j) \in \mathcal{I}_q(r)} \beta_{\rho,j|q} \boldsymbol{\mu}_{\rho,j}.\end{aligned}\tag{17}$$

Here, $\beta_{\rho,j|q}$ is the normalized attention weight for incidence (ρ, j) , satisfying $\sum_{(\rho,j) \in \mathcal{I}_q(r)} \beta_{\rho,j|q} = 1$ at each receiving relation node; ρ' and j' index the rules and their roles, respectively, in the normalization sum; $\boldsymbol{\mu}_{\rho,j} \in \mathbb{R}^d$ is the message generated by rule ρ for role j ; $\mathbf{m}_{r|q} \in \mathbb{R}^d$ is the aggregated rule message for relation node r conditioned on query relation q . Normalization is performed over all rule-role pairs incident to the same receiving relation node under the same query relation. When a relation occurs at multiple positions within a rule, each occurrence is treated as a separate incidence during attention normalization and message aggregation, preserving the contributions of different roles to the relation update. Both head and body relations receive rule information through their respective role-labeled incidences.

Gated Relation Update The aggregated message provides complementary information from complete rules for the current relation, while the base relation representation retains its local structural context. To combine these two sources of information, we introduce an element-wise update gate:

$$\begin{aligned}\mathbf{g}_{r|q} &= \sigma(\mathbf{W}_g[\mathbf{z}_{r|q} \parallel \mathbf{m}_{r|q}] + \mathbf{b}_g), & \mathcal{I}_q(r) \neq \emptyset, \\ \tilde{\mathbf{z}}_{r|q} &= \begin{cases} \text{LN}_u(\mathbf{z}_{r|q} + \mathbf{g}_{r|q} \odot \mathbf{m}_{r|q}), & \mathcal{I}_q(r) \neq \emptyset, \\ \mathbf{z}_{r|q}, & \mathcal{I}_q(r) = \emptyset. \end{cases}\end{aligned}\tag{18}$$

Here, $\mathbf{W}_g \in \mathbb{R}^{d \times 2d}$ and $\mathbf{b}_g \in \mathbb{R}^d$ are the gating parameters, $\mathbf{g}_{r|q} \in (0, 1)^d$ controls the extent to which rule messages are incorporated into each feature dimension, $\odot$ denotes element-wise multiplication, LN_u is the layer normalization used in the relation update, and $\tilde{\mathbf{z}}_{r|q}$ is the final enhanced relation representation. The residual term preserves the context provided by the base relation graph, while the gated term introduces complete rule information relevant to the current relation. Relations that do not participate in any selected rule retain their original representations; when $\mathcal{S}_q = \emptyset$, all relations retain the outputs of the base relation encoder. The resulting representations $\tilde{\mathbf{z}}_{r|q}$ serve as both edge relation and query relation representations in entity reasoning, allowing relation-level rule information to influence message passing between individual entities.

3.5 Rule-Enhanced Entity-Level Reasoning

As shown in Figure 3(d), given a query $(h, q, ?)$, QORH uses the enhanced relation representations to perform L_E layers of message passing over $\tilde{\mathcal{F}}$, starting

from the head entity h . Let $\mathcal{A}^{(\ell)}$ denote the set of active entities at layer ℓ , defined as

$$\mathcal{A}^{(0)} = \{h\}, \quad \mathcal{A}^{(\ell)} = \left\{ v \in \mathcal{V} \mid \exists u \in \mathcal{A}^{(\ell-1)}, r \in \overline{\mathcal{R}} : (u, r, v) \in \tilde{\mathcal{F}} \right\}. \quad (19)$$

Here, $\ell = 1, \dots, L_E$, and entity self-loops ensure that previously active nodes continue to participate in message passing. Let $\mathbf{h}_v^{(\ell)} \in \mathbb{R}^d$ denote the hidden state of entity v at layer ℓ ; we omit its conditioning subscript (h, q) for brevity and set $\mathbf{h}_h^{(0)} = \mathbf{0}_d$. For each edge (u, r, v) with $u \in \mathcal{A}^{(\ell-1)}$, we compute its message and query-conditioned propagation weight from the source entity state and the enhanced relation representations:

$$\begin{aligned} \mathbf{m}_{u,r,v}^{(\ell)} &= \mathbf{h}_u^{(\ell-1)} + \tilde{\mathbf{z}}_{r|q}, \\ \mathbf{a}_{u,r|q}^{(\ell)} &= \text{ReLU}\left(\mathbf{W}_s^{(\ell)} \mathbf{h}_u^{(\ell-1)} + \mathbf{W}_r^{(\ell)} \tilde{\mathbf{z}}_{r|q} + \mathbf{W}_q^{(\ell)} \tilde{\mathbf{z}}_{q|q} + \mathbf{b}_a^{(\ell)}\right), \\ \gamma_{u,r,v}^{(\ell)} &= \sigma\left(\left(\mathbf{w}_\alpha^{(\ell)}\right)^\top \mathbf{a}_{u,r|q}^{(\ell)} + b_\alpha^{(\ell)}\right). \end{aligned} \quad (20)$$

Here, $\mathbf{m}_{u,r,v}^{(\ell)} \in \mathbb{R}^d$ is the message on an entity-level edge, $\mathbf{a}_{u,r|q}^{(\ell)} \in \mathbb{R}^{d_a}$ is the query-conditioned attention feature, and $\gamma_{u,r,v}^{(\ell)} \in (0, 1)$ is the propagation weight of that edge; the matrices $\mathbf{W}_s^{(\ell)}, \mathbf{W}_r^{(\ell)}, \mathbf{W}_q^{(\ell)} \in \mathbb{R}^{d_a \times d}$ map the source entity state, edge relation representation, and query relation representation, respectively, $\mathbf{b}_a^{(\ell)}, \mathbf{w}_\alpha^{(\ell)} \in \mathbb{R}^{d_a}$ and $b_\alpha^{(\ell)} \in \mathbb{R}$ are bias and output parameters, and d_a is the hidden dimension of attention over entity-level edges; the edge weights $\gamma_{u,r,v}^{(\ell)} \in (0, 1)$ are computed independently using the sigmoid function, allowing multiple query-relevant edges to transmit information simultaneously.

The target entity then aggregates messages from nodes active at the preceding layer and updates its state using a GRU operating across layers:

$$\bar{\mathbf{h}}_v^{(\ell)} = \varphi\left(\mathbf{W}_h^{(\ell)} \sum_{\substack{u \in \mathcal{A}^{(\ell-1)} \\ (u,r,v) \in \tilde{\mathcal{F}}}} \gamma_{u,r,v}^{(\ell)} \mathbf{m}_{u,r,v}^{(\ell)}\right), \quad (21)$$

$$\mathbf{h}_v^{(\ell)} = \text{GRU}_E\left(\text{Dropout}\left(\bar{\mathbf{h}}_v^{(\ell)}\right), \hat{\mathbf{h}}_v^{(\ell-1)}\right). \quad (22)$$

Here, $\mathbf{W}_h^{(\ell)} \in \mathbb{R}^{d \times d}$ is a learnable transformation, and φ is an element-wise activation function. For previously active entities, $\hat{\mathbf{h}}_v^{(\ell-1)}$ is their hidden state from the preceding layer, whereas it is the zero vector for entities newly activated at the current layer; GRU_E shares parameters across entity reasoning layers and is parameterized independently of the rule body encoder GRU_B , while Dropout provides regularization during training.

After L_E propagation layers, a shared learnable scoring vector $\mathbf{w}_o \in \mathbb{R}^d$ is used to compute candidate tail entity scores:

$$s_\theta(h, q, t \mid \mathcal{G}_{\text{obs}}) = \begin{cases} \mathbf{w}_o^\top \mathbf{h}_t^{(L_E)}, & t \in \mathcal{A}^{(L_E)}, \\ 0, & t \notin \mathcal{A}^{(L_E)}. \end{cases} \quad (23)$$

3.6 Optimization and Inductive Inference

The entity-inductive and fully inductive experiments for QORH follow the data splits and evaluation protocols of GraIL [13] and InGram [18], respectively. During training, we partition the training facts into message-passing facts and supervision queries and update the relation dependency graph and rule base as the observed facts change. Model parameters are optimized exclusively on training data, with the validation set used for model selection and the test set reserved for final evaluation. Given a batch of supervision triples $\mathcal{D}_b$, we apply softmax normalization to the scores over the set of training candidate entities $\mathcal{V}_{\text{tr}}$ and minimize the negative log-likelihood loss:

$$p_\theta(t \mid h, q, \mathcal{G}_{\text{obs}}) = \frac{\exp(s_\theta(h, q, t \mid \mathcal{G}_{\text{obs}}))}{\sum_{v \in \mathcal{V}_{\text{tr}}} \exp(s_\theta(h, q, v \mid \mathcal{G}_{\text{obs}}))}, \quad (24)$$

$$\mathcal{L}(\theta) = - \sum_{(h, q, t) \in \mathcal{D}_b} \log p_\theta(t \mid h, q, \mathcal{G}_{\text{obs}}).$$

Here, p_θ is the conditional predictive probability of a candidate tail entity, t is the correct tail entity, v ranges over all training candidate entities, and $\mathcal{L}$ is the batch loss. All candidate entities are included in the normalization, eliminating the need for additional negative sampling. During inductive inference, with the parameters θ fixed, we reconstruct the relation graph, mine rules, and compute their statistics from the observed facts in the inference graph before scoring the candidate entities in $\mathcal{V}_{\text{inf}}$. Because the shared parameters are independent of specific entity or relation IDs, the model can generate representations for unseen entities and relations from the structure of a new graph.

To avoid redundant computation, queries with the same query relation within a batch share the relation encoding and hypergraph enhancement results; candidate rules and their statistics can also be reused when the observed graph remains unchanged. For each query relation, at most K rules with body length no greater than L_{max} are selected, and the total number of incidences satisfies $|\mathcal{I}_q| = \sum_{\rho \in \mathcal{S}_q} (\ell_\rho + 1) \leq K(L_{\text{max}} + 1)$, bounding the scale of hypergraph propagation through the limits on rule count and length.

4 Experiments

This section evaluates QORH on entity-inductive and fully inductive knowledge graph link prediction tasks, covering 12 datasets in each setting. Using mean reciprocal rank (MRR) as the primary evaluation metric, QORH achieves the best performance among the compared methods on 11 of the 12 entity-inductive datasets and all 12 fully inductive datasets, demonstrating its strong link prediction performance in both inductive settings.

4.1 Datasets and Evaluation Metrics

Entity-Inductive Datasets We adopt the entity-inductive link prediction benchmarks introduced by GraIL [13]. These benchmarks comprise three dataset families derived from WN18RR [27], FB15k-237 [28], and NELL-995 [33], each containing four versions, v1 through v4, yielding 12 datasets in total. Each version consists of a training graph and a corresponding inductive inference graph, enabling evaluation of generalization across different graph sizes and relational structures. Under the benchmark protocol, the training and inference graphs must have disjoint entity sets, while inference uses the relation vocabulary available during training, i.e., $\mathcal{V}_{\text{tr}} \cap \mathcal{V}_{\text{inf}} = \emptyset$, $\mathcal{R}_{\text{inf}} \subseteq \mathcal{R}_{\text{tr}}$.

Fully Inductive Datasets We further adopt the benchmarks introduced by InGram [18], which comprise three dataset families, FB, NL, and WK, derived from FB15k-237 [28], NELL-995 [33], and Wikidata68K [34], respectively. Each family includes four settings, 25, 50, 75, and 100, yielding 12 datasets in total. These numbers specify the nominal proportions of triples involving unseen relations during dataset construction, rather than the proportions of unseen relation types; the actual proportions in the data files may deviate from these nominal values. In the first three settings, the inference graphs contain both seen and unseen relations, whereas the 100 setting enforces completely disjoint relation sets between the training and inference graphs. Following the task definition in Section 3.1, we collectively refer to these settings that allow unseen relations as fully inductive settings. Each dataset contains `train.txt`, `msg.txt`, `valid.txt`, and `test.txt`. The `train.txt` file is used for model training. The remaining three files belong to an inference graph whose entities are disjoint from those on the training side and provide observed facts, validation queries, and test queries, respectively.

Evaluation Metrics We evaluate link prediction performance using mean reciprocal rank (MRR), Hits@1, and Hits@10. For each test triple, we perform both head and tail entity prediction. Head prediction is converted into tail prediction using inverse relations, consistent with the inference formulation in Section 3. Let N denote the total number of prediction instances evaluated in both directions, and let rank_i denote the rank of the target entity for the i -th instance. The metrics are defined as follows:

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^N \frac{1}{\text{rank}_i}, \quad (25)$$

$$\text{Hits}@k = \frac{1}{N} \sum_{i=1}^N \mathbb{I}(\text{rank}_i \leq k), \quad k \in \{1, 10\}, \quad (26)$$

where $\mathbb{I}(\cdot)$ is the indicator function. MRR measures the ranking positions of correct answers and is particularly sensitive to changes near the top of the candidate

list. Hits@1 and Hits@10 measure the proportions of correct answers ranked first and within the top ten, respectively. Higher values indicate better performance for all three metrics.

To comprehensively evaluate ranking performance over the candidate entity space, we use filtered evaluation with all entities as candidates. For a query $(h, q, ?)$, all entities in the current message-passing graph are considered as candidates and ranked according to $s_\theta(h, q, t \mid \mathcal{G}_{\text{obs}})$. When computing the target entity’s rank, we filter out other candidate entities that form known true triples with the query, preventing valid alternative answers from being counted as incorrect predictions. Accordingly, no negative sampling is performed during evaluation. Compared with the original experimental settings of GraIL [13], MorsE [15], and related methods, which rank each test triple against 50 randomly sampled negative triples, all-entity ranking covers the complete candidate space and avoids the randomness introduced by candidate sampling. All MRR, Hits@1, and Hits@10 scores reported for QORH are computed under this all-entity evaluation setting.

4.2 Baselines and Experimental Setup

To systematically evaluate QORH on entity-inductive and fully inductive link prediction tasks, we select 20 representative baseline methods spanning five methodological categories: logical rule learning, entity-graph reasoning, structural representation learning, semantic feature utilization, and relational structure transfer.

- (1) **Logical rule learning methods:** RuleN [35], Neural LP [22], and DRUM [23]. These methods learn logical composition patterns among relations and predict missing facts through rule instantiation.
- (2) **Entity-graph structure and path propagation methods:** GraIL [13], CoMPiLE [36], SNRI [37], INDIGO [38], NBFNet [25], RED-GNN [26], and Context Pooling [16]. These methods extract structural evidence from entity neighborhoods, local subgraphs, or multi-hop paths to learn transferable link prediction patterns.
- (3) **Representation methods based on structural encoding and meta-learning:** CompGCN [39], NodePiece [40], and MorsE [15]. These methods learn representations for link prediction through compositional encoding of entities and relations, shared representation units, or transferable mechanisms for generating embeddings.
- (4) **Methods using semantic and auxiliary features:** BLP [41], QBLP [42], and RAILD [34]. These methods incorporate textual descriptions, entity features, or fact qualifiers to provide additional semantic evidence for inductive prediction.
- (5) **Relational structure transfer methods:** RMPI [17], InGram [18], TARGI [19], and GraphOracle [21]. These methods explicitly model structural connections among relations and use relation-level message passing to support generalization to new entities and relations.

Experiments are conducted on a server running Ubuntu 20.04 with an NVIDIA Tesla V100 GPU equipped with 32 GB of memory. QORH is implemented in PyTorch and uses `torch_scatter` for sparse message aggregation. We use the Adam optimizer and set the random seed to 42. The hidden representation dimension is selected from $\{16, 32, 48, 64\}$, the number of entity-level message-passing layers from 3, 5, and the batch size from 10, 20, 24, 100. The initial learning rate, L_2 regularization coefficient, and dropout rate range over 1.6×10^{-4} , 9.2×10^{-3} , 1.8×10^{-5} , 4.0×10^{-4} , and 0.02, 0.30, respectively. For the ordered rule hypergraph module, the maximum rule length is selected from $\{2, 3\}$, the minimum rule support is set to 2, and the number of rules selected for each query relation is chosen from $\{1, 2, 3, 4, 5, 6, 7\}$. QORH is trained for a maximum of 200 epochs, with validation MRR used for model selection and early stopping. Training terminates when validation MRR fails to improve for 10 consecutive epochs, and the model with the highest validation MRR is used for final testing.

As discussed in Sections 2.2 and 4.2, the pretraining data of existing knowledge graph foundation models overlap with inductive test data, which is inconsistent with the inductive evaluation protocol. As shown in Table 1, the official pretraining pipeline of GraphOracle [21] includes source knowledge graphs such as FB15k-237 [28] and WN18RR [27], from which some inductive benchmarks are also constructed. Consequently, even under zero-shot evaluation on the target dataset, a substantial fraction of the test entities and relations have already been encountered by the pretrained model.

To eliminate the effects of this data overlap, we strictly follow the official GraphOracle [21] implementation (<https://github.com/EnjunDu/GraphOracle>) and adapt its data loading, training, and testing procedures to the entity-inductive protocol introduced by GraIL [13] and the fully inductive protocol introduced by InGram [18]. Specifically, consistent with the other baselines, GraphOracle [21] is trained independently on the designated training graph of each benchmark, with model selection based on the corresponding validation set. During testing, model parameters are fixed, observed facts in the corresponding inference graph are used for message passing, and prediction metrics are computed according to the evaluation procedure in Section 4.1.

4.3 Main Results

Tables 2, 3, 4, and 5 report the comparative results of QORH and the baselines on entity-inductive and fully inductive link prediction tasks. As shown in Table 2, QORH achieves the highest MRR and Hits@1 on 11 of the 12 entity-inductive datasets and the highest Hits@10 on all 12 datasets. For example, on FB15k-237-v1, QORH improves MRR from 0.383, achieved by Context Pooling [16], to 0.466, corresponding to a relative improvement of 21.67%. This result demonstrates its performance advantage over existing query-aware graph reasoning methods.

Tables 3, 4, 5 present the fully inductive results for the FB, NL, and WK families, respectively. QORH achieves the highest MRR and Hits@1 on all 12 fully inductive datasets and the highest Hits@10 on 10 of them. In particular, on FB-100, NL-100, and WK-100, both the entity sets and the relation sets are

Table 1. Overlap between GraphOracle’s pretraining data and selected inductive test sets (%).

Inductive setting	Dataset	Entity overlap	Relation overlap
Entity-inductive	FB15k-237-v1	100.00	—
	FB15k-237-v2	96.98	—
	FB15k-237-v3	98.57	—
	FB15k-237-v4	98.88	—
	WN18RR-v1	83.92	—
	WN18RR-v2	81.41	—
	WN18RR-v3	71.18	—
	WN18RR-v4	82.03	—
Fully inductive	FB-25	97.34	97.45
	FB-50	98.25	97.66
	FB-75	95.45	97.35
	FB-100	95.78	100.00

Note: Entity and relation overlap indicate the percentages of distinct entities and relations in the inductive test set that appear in GraphOracle’s pretraining data. GraphOracle’s released data loader constructs training facts from `train.txt` without reading `facts.txt`, so overlap may be below 100%. “—” denotes unreported relation overlap in the entity-inductive setting, which uses the training relation vocabulary.

completely disjoint between the training and inference graphs. This entity- and relation-disjoint setting requires the model to generalize simultaneously to unseen entities and relations, making it more challenging. On these three datasets, QORH achieves MRR scores of 0.456, 0.462, and 0.181, respectively, outperforming the best baseline on each dataset by 20.95%, 47.13%, and 30.22%. These results demonstrate that QORH can effectively exploit structural information in the inference graph to predict missing links even when no entities or relations are shared between training and inference, further supporting its inductive generalization capability under simultaneous changes in entities and relations.

4.4 Ablation Studies

To investigate the contributions of ordered rule hypergraph enhancement and rule selection, we conduct ablation studies on the 12 entity-inductive datasets using two variants. QORH w/o RHE removes the entire rule hypergraph enhancement module while retaining the base relation encoding and entity-level reasoning. QORH w/o RS removes the rule selection module and uses all candidate rules for hypergraph enhancement while retaining ordered rule encoding and hypergraph message passing. Detailed results are reported in Table 6, where the averages are computed as the unweighted means across the 12 dataset variants.

Table 2. Entity-inductive link prediction results on WN18RR, FB15k-237, and NELL-995. Bold and underlined values indicate the best and second-best results, respectively.

Metric	Models	WN18RR				FB15k-237				NELL-995			
		V1	V2	V3	V4	V1	V2	V3	V4	V1	V2	V3	V4
MRR	RuleN*	0.668	0.645	0.368	0.624	0.363	0.433	0.439	0.429	0.615	0.385	0.381	0.333
	Neural LP*	0.649	0.635	0.361	0.628	0.325	0.389	0.400	0.396	0.610	0.361	0.367	0.261
	DRUM [†]	0.666	0.646	0.380	0.627	0.333	0.395	0.402	0.410	0.628	0.365	0.375	0.273
	GraIL*	0.627	0.625	0.323	0.553	0.279	0.276	0.251	0.227	0.481	0.297	0.322	0.262
	CoMPILE*	0.577	0.578	0.308	0.548	0.287	0.276	0.262	0.213	0.330	0.248	0.319	0.229
	NBFNet*	0.684	0.652	0.425	0.604	0.307	0.369	0.331	0.305	0.584	0.410	0.425	0.287
	MorsE [#]	0.429	0.389	0.229	0.324	0.228	0.298	0.265	0.232	0.334	0.111	0.099	0.046
	RED-GNN*	0.701	0.690	<u>0.427</u>	<u>0.651</u>	0.369	<u>0.469</u>	0.445	<u>0.442</u>	0.637	0.419	<u>0.436</u>	0.363
	InGram [#]	0.279	0.235	0.114	0.230	0.192	0.223	0.188	0.162	0.672	0.219	0.278	0.203
	Context Pooling*	<u>0.708</u>	<u>0.694</u>	0.425	0.648	<u>0.383</u>	0.463	<u>0.447</u>	0.427	0.634	<u>0.430</u>	0.426	<u>0.382</u>
	GraphOracle [^]	0.656	0.634	0.393	0.490	0.362	0.307	0.407	0.219	0.504	0.360	0.404	0.262
	QORH	0.793	0.725	0.490	0.679	0.466	0.533	0.517	0.497	<u>0.649</u>	0.611	0.567	0.535
Hits@1	RuleN*	0.635	0.611	0.347	0.592	0.309	0.347	0.345	0.338	0.545	0.304	0.303	0.248
	Neural LP*	0.592	0.575	0.304	0.583	0.243	0.286	0.309	0.289	0.500	0.249	0.267	0.137
	DRUM*	0.613	0.595	0.330	0.586	0.247	0.284	0.308	0.309	0.500	0.271	0.262	0.163
	GraIL*	0.554	0.542	0.278	0.443	0.205	0.202	0.165	0.143	0.425	0.199	0.224	0.153
	CoMPILE*	0.473	0.485	0.258	0.473	0.208	0.178	0.166	0.134	0.105	0.156	0.226	0.159
	NBFNet*	0.592	0.575	0.304	0.574	0.190	0.229	0.206	0.185	0.500	0.271	0.262	0.233
	MorsE [#]	0.343	0.320	0.179	0.244	0.117	0.168	0.146	0.128	0.235	0.046	0.040	0.014
	RED-GNN*	0.653	0.633	<u>0.368</u>	<u>0.606</u>	0.302	<u>0.381</u>	0.351	<u>0.340</u>	0.525	0.319	<u>0.345</u>	0.259
	InGram [#]	0.146	0.125	0.056	0.100	0.093	0.127	0.106	0.087	0.570	0.113	0.189	0.125
	Context Pooling*	<u>0.660</u>	<u>0.637</u>	0.365	0.603	<u>0.316</u>	0.368	<u>0.358</u>	0.330	0.517	<u>0.329</u>	0.324	<u>0.278</u>
	GraphOracle [^]	0.516	0.535	0.326	0.289	0.273	0.189	0.293	0.116	0.455	0.228	0.326	0.131
	QORH	0.753	0.678	0.423	0.638	0.381	0.443	0.425	0.399	<u>0.550</u>	0.507	0.466	0.415
Hits@10	RuleN*	0.730	0.694	0.407	0.681	0.446	0.599	0.600	0.605	0.760	0.514	0.531	0.484
	Neural LP*	0.772	0.749	0.476	0.706	0.468	0.586	0.571	0.593	0.871	0.564	0.576	0.539
	DRUM*	0.777	0.747	0.477	0.702	0.474	0.595	0.571	0.593	<u>0.873</u>	0.540	0.577	0.531
	GraIL*	0.760	0.776	0.409	0.687	0.429	0.424	0.424	0.389	0.565	0.496	0.518	0.506
	CoMPILE*	0.747	0.743	0.406	0.670	0.439	0.457	0.449	0.358	0.575	0.446	0.515	0.421
	NBFNet*	<u>0.827</u>	<u>0.799</u>	<u>0.563</u>	0.702	0.517	0.639	0.588	0.559	0.795	0.635	0.606	<u>0.591</u>
	MorsE [#]	0.588	0.523	0.307	0.469	0.471	0.582	0.504	0.454	0.485	0.241	0.213	0.096
	RED-GNN*	0.799	0.780	0.524	0.721	0.483	0.629	0.603	<u>0.621</u>	0.866	0.601	0.594	0.556
	InGram [#]	0.524	0.449	0.225	0.456	0.373	0.426	0.340	0.311	0.845	0.423	0.464	0.365
	Context Pooling [#]	0.790	0.785	0.527	<u>0.723</u>	0.473	<u>0.640</u>	0.589	0.592	0.866	<u>0.636</u>	<u>0.619</u>	0.564
	GraphOracle [^]	0.798	0.783	0.523	0.700	<u>0.517</u>	0.536	<u>0.611</u>	0.428	0.535	0.616	0.541	0.563
	QORH	0.878	0.801	0.603	0.752	0.600	0.694	0.677	0.676	0.950	0.792	0.745	0.741

* Results taken directly from [16] or [21].

[#] Results reproduced using the code released in the respective official repositories, following the training and testing protocols for the entity-inductive setting.[^] Results reproduced by adapting the code released in the respective official repositories to the entity-inductive training and testing protocols, with modifications to the data loading, training, and evaluation procedures. See the *Baselines and Experiment Setup* section for details.

Table 3. Performance comparison on FB-100, FB-75, FB-50, and FB-25 under the fully inductive setting. The best and second-best results are highlighted in bold and underlined, respectively.

Model	FB-100			FB-75			FB-50			FB-25		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
CompGCN*	0.015	0.008	0.025	0.013	0.000	0.026	0.004	0.002	0.006	0.003	0.000	0.004
NodePiece*	0.006	0.001	0.009	0.016	0.007	0.029	0.021	0.006	0.048	0.044	0.011	0.114
NeuralLP*	0.026	0.007	0.057	0.056	0.030	0.099	0.088	0.043	0.184	0.164	0.098	0.309
DRUM*	0.034	0.011	0.077	0.065	0.034	0.121	0.101	0.061	0.191	0.175	0.109	0.320
BLP*	0.017	0.004	0.035	0.047	0.024	0.085	0.078	0.037	0.156	0.107	0.053	0.212
QBLP*	0.013	0.003	0.026	0.041	0.017	0.084	0.071	0.030	0.147	0.104	0.043	0.226
NBFNet*	0.072	0.026	0.154	0.089	0.048	0.166	0.130	0.071	0.259	0.224	0.137	0.410
RED-GNN*	0.121	0.053	0.263	0.107	0.057	0.201	0.129	0.072	0.251	0.145	0.077	0.284
RAILD*	0.031	0.016	0.048	–	–	–	–	–	–	–	–	–
INGRAM*	0.223	0.146	0.371	0.189	0.119	0.325	0.117	0.067	0.218	0.133	0.067	0.271
Context Pooling [#]	0.146	0.056	0.323	0.128	0.074	0.242	0.144	0.069	0.307	0.235	0.142	0.425
TARGI*	0.253	0.172	0.429	–	–	–	0.159	0.088	0.307	–	–	–
GraphOracle [^]	<u>0.377</u>	<u>0.277</u>	<u>0.573</u>	<u>0.315</u>	<u>0.215</u>	<u>0.510</u>	<u>0.190</u>	<u>0.107</u>	<u>0.365</u>	<u>0.276</u>	<u>0.164</u>	<u>0.513</u>
QORH	0.456	0.358	0.651	0.401	0.295	0.597	0.322	0.224	0.522	0.376	0.255	0.625

* Results are taken directly from [18], [21], or [19].

[#] Results are reproduced using the official code under the training and evaluation protocols for the fully inductive setting.

[^] The corresponding result is not available.

[^] Results are reproduced using the official code, with the data loading, training, and evaluation procedures adapted to the fully inductive training and evaluation protocols. See the *Baselines and Experiment Setup* section for details.

Removing rule hypergraph enhancement degrades all three metrics on all 12 datasets. Average MRR, Hits@1, and Hits@10 decrease from 0.5885, 0.5064, and 0.7423 to 0.4634, 0.3812, and 0.6195, respectively, with average MRR dropping by 21.25% relative to the full model. These results demonstrate the effectiveness of rule hypergraph enhancement for inductive reasoning. The base relation graph captures pairwise dependencies but omits higher-order joint information involving multiple relations.

Removing rule selection reduces average MRR, Hits@1, and Hits@10 to 0.5521, 0.4617, and 0.7095, corresponding to relative decreases of 6.17%, 8.83%, and 4.41% compared with the full model. These results indicate that directly incorporating all candidate rules does not yield consistent performance gains. This trend is consistent with the rationale for selecting rules based on confidence, coverage, rule length, and support, indicating that rule selection improves the usefulness of the incorporated evidence.

The ablation results show that rule hypergraph enhancement and rule selection serve distinct roles: the ordered rule hypergraph provides information complementary to the base relation encoding, while rule selection further improves the utilization of this evidence.

Table 4. Performance comparison on NELL-100, NELL-75, NELL-50, and NELL-25 under the fully inductive setting. The best and second-best results are highlighted in bold and underlined, respectively.

Model	NL-100			NL-75			NL-50			NL-25		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
GraLL [*]	0.135	0.114	0.173	0.096	0.056	0.205	0.162	0.104	0.288	0.216	0.160	0.366
CoMPILE [*]	0.123	0.071	0.209	0.178	0.093	0.361	0.194	0.125	0.330	0.189	0.115	0.324
SNRI [*]	0.042	0.029	0.064	0.088	0.040	0.177	0.130	0.095	0.187	0.190	0.140	0.270
INDIGO [*]	0.160	0.109	0.247	0.121	0.098	0.156	0.167	0.134	0.217	0.166	0.134	0.206
RMPI [*]	0.220	0.136	0.376	0.138	0.061	0.275	0.185	0.109	0.307	0.213	0.130	0.329
CompGCN [*]	0.008	0.001	0.014	0.014	0.003	0.025	0.003	0.000	0.005	0.006	0.000	0.010
NodePiece [*]	0.012	0.004	0.018	0.042	0.020	0.081	0.037	0.013	0.079	0.098	0.057	0.166
NeuralLP [*]	0.084	0.035	0.181	0.117	0.048	0.273	0.101	0.064	0.190	0.148	0.101	0.271
DRUM [*]	0.076	0.044	0.138	0.152	0.072	0.313	0.107	0.070	0.193	0.161	0.119	0.264
BLP [*]	0.019	0.006	0.037	0.051	0.012	0.120	0.041	0.011	0.093	0.049	0.024	0.095
QBLP [*]	0.004	0.000	0.003	0.040	0.007	0.095	0.048	0.020	0.097	0.073	0.027	0.151
NBFNet [*]	0.096	0.032	0.199	0.137	0.077	0.255	0.225	0.161	0.346	0.283	0.224	0.417
RED-GNN [*]	0.212	0.114	0.385	0.203	0.129	0.353	0.179	0.115	0.280	0.214	0.166	0.266
RAILD [*]	0.018	0.005	0.037	—	—	—	—	—	—	—	—	—
INGRAM [*]	0.309	0.212	<u>0.506</u>	<u>0.261</u>	<u>0.167</u>	<u>0.464</u>	0.281	0.193	0.453	<u>0.334</u>	<u>0.241</u>	0.501
Context Pooling [#]	0.219	0.138	0.370	0.180	0.099	0.336	0.226	0.146	0.363	0.231	0.170	0.307
TARGI [*]	<u>0.314</u>	<u>0.230</u>	0.489	—	—	—	<u>0.314</u>	<u>0.207</u>	0.533	—	—	—
GraphOracle [^]	0.247	0.149	0.435	0.244	0.155	0.424	0.255	0.169	0.420	0.261	0.173	0.400
QORH	0.462	0.363	0.644	0.336	0.245	0.536	0.380	0.303	<u>0.519</u>	0.365	0.294	<u>0.475</u>

^{*} Results are taken directly from [18], [21], or [19].

[#] Results are reproduced using the official code under the training and evaluation protocols for the fully inductive setting.

⁻ The corresponding result is not available.

[^] Results are reproduced using the official code, with the data loading, training, and evaluation procedures adapted to the fully inductive training and evaluation protocols. See the *Baselines and Experiment Setup* section for details.

4.5 Hyperparameter Sensitivity Analysis

To assess the sensitivity of the ordered rule hypergraph module to its key hyperparameters, we examine the maximum number of rules per query relation K and the maximum rule-body length $L_{\max}$ in both entity-inductive and fully inductive settings. With $K \in \{1, 2, 3, 4, 5, 6, 7\}$ and $L_{\max} \in \{2, 3\}$, we compare 14 parameter combinations on each dataset.

The effect of rule length varies across datasets. As shown in Figure 6, in the fully inductive setting, $L_{\max} = 3$ yields modest improvements over $L_{\max} = 2$ on FB-25, FB-50, and WK-100, with a more pronounced improvement on NL-100. As shown in Figure 4, in the entity-inductive setting, $L_{\max} = 3$ provides slight gains on WN18RR-v3 and FB15k-237-v2 and a more substantial gain on NELL-995-v4. Accordingly, we use $L_{\max} = 3$ for these datasets and $L_{\max} = 2$ for the remaining datasets.

As shown in Figures 5 and 7, performance varies only slightly across different values of K in the fully inductive setting, indicating that the model is relatively insensitive to this hyperparameter. Across both inductive settings, $K = 4$ provides consistently robust performance across datasets. We therefore set $K = 4$ for

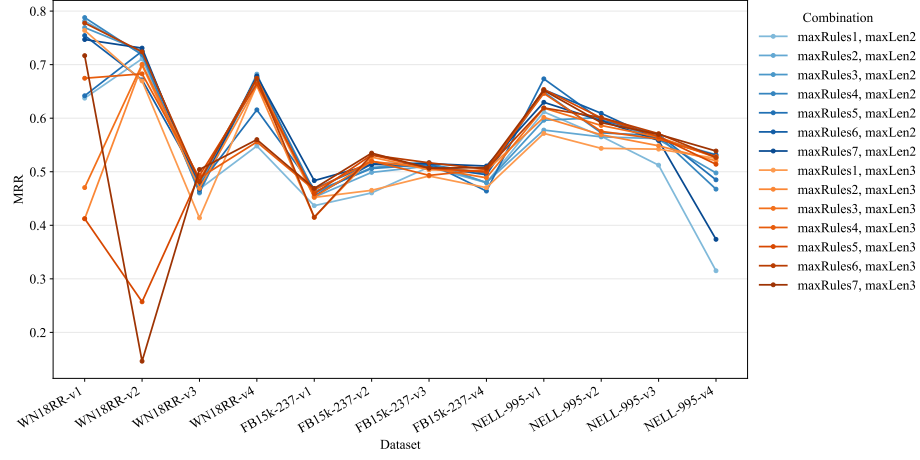


Fig. 4. MRR of hyperparameter combination across entity-inductive datasets.

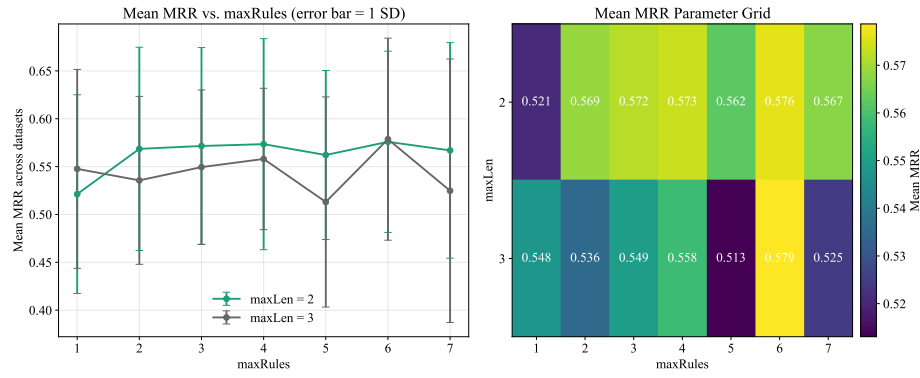


Fig. 5. Hyperparameter sensitivity summary on entity-inductive Datasets.

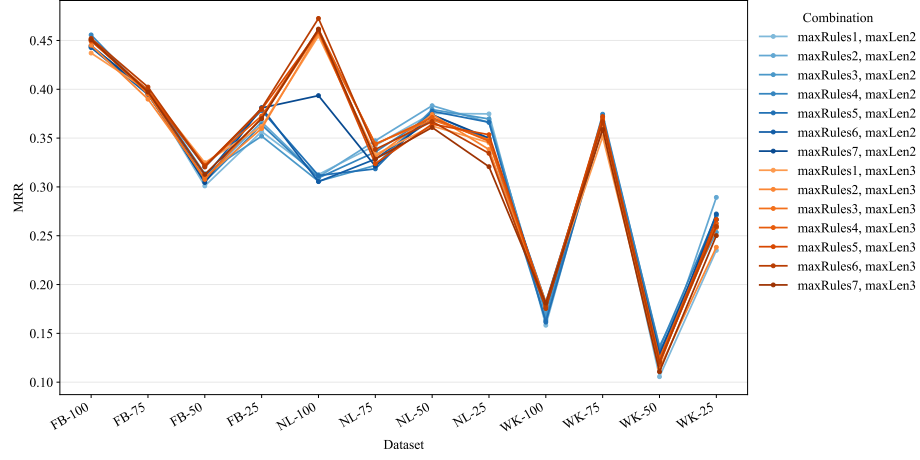


Fig. 6. MRR of hyperparameter combination across fully inductive datasets.

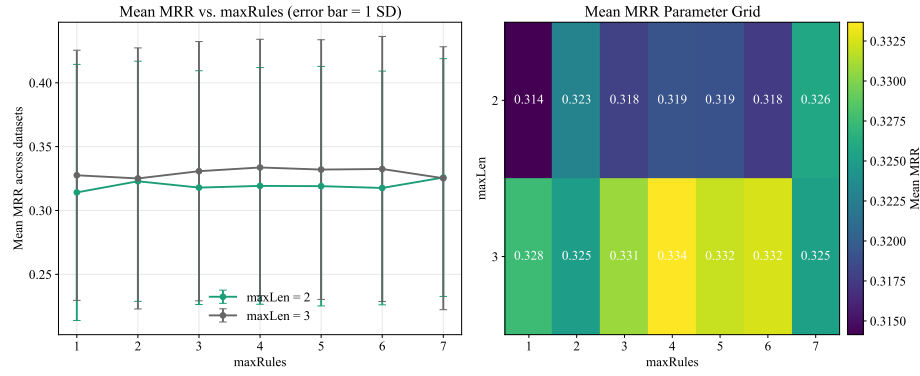


Fig. 7. Hyperparameter sensitivity summary on fully inductive Datasets.

Table 5. Performance comparison on WK-100, WK-75, WK-50, and WK-25 under the fully inductive setting. The best and second-best results are highlighted in bold and underlined, respectively.

Model	WK-100			WK-75			WK-50			WK-25		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
CompGCN [*]	0.003	0.000	0.009	0.015	0.003	0.028	0.003	0.001	0.002	0.009	0.000	0.020
NodePiece [*]	0.007	0.002	0.018	0.021	0.003	0.052	0.008	0.002	0.013	0.053	0.019	0.122
NeuralLP [*]	0.009	0.005	0.016	0.020	0.004	0.054	0.025	0.007	0.054	0.068	0.046	0.104
DRUM [*]	0.010	0.004	0.019	0.020	0.007	0.043	0.017	0.002	0.046	0.064	0.035	0.116
BLP [*]	0.012	0.003	0.025	0.043	0.016	0.089	0.041	0.013	0.092	0.125	0.055	0.283
QBLP [*]	0.012	0.003	0.025	0.044	0.016	0.091	0.035	0.011	0.080	0.116	0.042	0.294
NBFNet [*]	0.014	0.005	0.026	0.072	0.028	0.172	0.062	0.036	0.105	0.154	0.092	0.301
RED-GNN [*]	0.096	0.070	0.136	0.172	0.110	0.290	0.058	0.033	0.093	0.170	0.111	0.263
RAILD [*]	0.026	0.010	0.052	—	—	—	—	—	—	—	—	—
INGRAM [*]	0.107	0.072	0.169	<u>0.247</u>	<u>0.179</u>	0.362	<u>0.068</u>	0.034	<u>0.135</u>	<u>0.186</u>	<u>0.124</u>	<u>0.309</u>
Context Pooling [#]	0.117	0.082	0.185	0.181	0.100	0.322	0.032	0.011	0.065	0.151	0.078	0.282
GraphOracle [^]	<u>0.139</u>	<u>0.097</u>	<u>0.231</u>	0.204	0.123	<u>0.371</u>	0.057	<u>0.036</u>	0.096	0.132	0.050	0.278
QORH	0.181	0.128	0.282	0.375	0.291	0.513	0.135	0.080	0.243	0.276	0.195	0.429

^{*} Results are taken directly from [18] or [21].

[#] Results are reproduced using the official code under the training and evaluation protocols for the fully inductive setting.

[—] The corresponding result is not available.

[^] Results are reproduced using the official code, with the data loading, training, and evaluation procedures adapted to the fully inductive training and evaluation protocols. See the *Baselines and Experiment Setup* section for details.

all datasets. All QORH results reported in Section 4.3 use these hyperparameter configurations.

5 Conclusion

We propose QORH, a query-specific ordered rule hypergraph method, to address the difficulty of preserving complete rule composition information through pairwise relational dependencies in inductive knowledge graph link prediction. Conditioned on the query relation, QORH jointly encodes rule heads, ordered rule bodies, and statistical evidence as hyperedges. Through learnable rule selection and role-aware relation-hyperedge-relation propagation, it incorporates joint dependencies among multiple relations into relation representations, which subsequently guide entity-level reasoning. Extensive experiments in both entity-inductive and fully inductive settings show that QORH substantially outperforms existing state-of-the-art baselines, demonstrating its effectiveness and robustness. In future work, we will explore joint modeling of rule structures and textual semantics, investigate rule generation and selection assisted by large language models, and extend ordered rule hypergraphs to a broader range of structured reasoning tasks.

Table 6. Ablation results on the v1–v4 splits of WN18RR, FB15k-237, and NELL-995 under the entity-inductive setting. The best results are highlighted in bold.

Dataset	Version	QORH			w/o RHE			w/o RS		
		MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
WN18RR	v1	0.793	0.753	0.878	0.737	0.681	0.843	0.738	0.665	0.859
	v2	0.725	0.678	0.801	0.676	0.602	0.793	0.693	0.629	0.801
	v3	0.490	0.423	0.603	0.453	0.385	0.569	0.454	0.395	0.564
	v4	0.679	0.638	0.752	0.128	0.057	0.280	0.627	0.560	0.735
FB15k-237	v1	0.466	0.381	0.600	0.454	0.373	0.573	0.454	0.378	0.571
	v2	0.533	0.443	0.694	0.508	0.409	0.686	0.499	0.384	0.694
	v3	0.517	0.425	0.677	0.495	0.395	0.663	0.500	0.402	0.660
	v4	0.497	0.399	0.676	0.260	0.154	0.483	0.478	0.375	0.670
NELL-995	v1	0.649	0.550	0.950	0.573	0.505	0.780	0.679	0.535	0.945
	v2	0.611	0.507	0.792	0.489	0.374	0.704	0.597	0.498	0.766
	v3	0.567	0.466	0.745	0.546	0.453	0.712	0.549	0.454	0.706
	v4	0.535	0.415	0.741	0.243	0.186	0.350	0.358	0.266	0.546
Macro-average		0.588	0.506	0.742	0.463	0.381	0.620	0.552	0.462	0.710

Notes. w/o RHE and w/o RS denote QORH variants without rule hypergraph enhancement and learnable rule selection, respectively. H@1 and H@10 denote Hits@1 and Hits@10. Macro-average is the unweighted mean over the 12 dataset splits. For each split and metric, the best result across the three variants is shown in bold.

References

1. Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., Guo, M.: RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 417–426. CIKM ’18, Association for Computing Machinery, New York, NY, USA (Oct 2018). <https://doi.org/10.1145/3269206.3271739>
2. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: KGAT: Knowledge Graph Attention Network for Recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 950–958. KDD ’19, Association for Computing Machinery, New York, NY, USA (Jul 2019). <https://doi.org/10.1145/3292500.3330989>
3. Bordes, A., Chopra, S., Weston, J.: Question Answering with Subgraph Embeddings. In: Moschitti, A., Pang, B., Daelemans, W. (eds.) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 615–620. Association for Computational Linguistics, Doha, Qatar (Oct 2014). <https://doi.org/10.3115/v1/D14-1067>
4. Yih, W.t., Chang, M.W., He, X., Gao, J.: Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base. In: Zong, C.,

- Strube, M. (eds.) Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 1321–1331. Association for Computational Linguistics, Beijing, China (Jul 2015). <https://doi.org/10.3115/v1/P15-1128>
5. Belleau, F., Nolin, M.A., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. *Journal of Biomedical Informatics* **41**(5), 706–716 (Oct 2008). <https://doi.org/10.1016/j.jbi.2008.03.004>
 6. Himmelstein, D.S., Lizee, A., Hessler, C., Brueggeman, L., Chen, S.L., Hadley, D., Green, A., Khankhanian, P., Baranzini, S.E.: Systematic integration of biomedical knowledge prioritizes drugs for repurposing. *eLife* **6**, e26726 (Sep 2017). <https://doi.org/10.7554/eLife.26726>
 7. Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., Liu, Q.: ERNIE: Enhanced Language Representation with Informative Entities. In: Korhonen, A., Traum, D., Màrquez, L. (eds.) Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 1441–1451. Association for Computational Linguistics, Florence, Italy (Jul 2019). <https://doi.org/10.18653/v1/P19-1139>
 8. Liu, W., Zhou, P., Zhao, Z., Wang, Z., Ju, Q., Deng, H., Wang, P.: K-BERT: Enabling Language Representation with Knowledge Graph. Proceedings of the AAAI Conference on Artificial Intelligence **34**(03), 2901–2908 (Apr 2020). <https://doi.org/10.1609/aaai.v34i03.5681>
 9. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating Embeddings for Modeling Multi-relational Data. In: Advances in Neural Information Processing Systems. vol. 26. Curran Associates, Inc. (2013)
 10. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In: International Conference on Learning Representations (Dec 2014)
 11. Trouillon, T., Welbl, J., Riedel, S., Gaussier, E., Bouchard, G.: Complex Embeddings for Simple Link Prediction. In: Proceedings of The 33rd International Conference on Machine Learning. pp. 2071–2080. PMLR (Jun 2016)
 12. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In: International Conference on Learning Representations (Sep 2018)
 13. Teru, K.K., Denis, E.G., Hamilton, W.L.: Inductive relation prediction by subgraph reasoning. In: Proceedings of the 37th International Conference on Machine Learning. ICML’20, vol. 119, pp. 9448–9457. JMLR.org (Jul 2020)
 14. Chen, J., He, H., Wu, F., Wang, J.: Topology-Aware Correlations Between Relations for Inductive Link Prediction in Knowledge Graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 6271–6278 (May 2021). <https://doi.org/10.1609/aaai.v35i7.16779>
 15. Chen, M., Zhang, W., Zhu, Y., Zhou, H., Yuan, Z., Xu, C., Chen, H.: Meta-Knowledge Transfer for Inductive Knowledge Graph Embedding. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 927–937. SIGIR ’22, Association for Computing Machinery, New York, NY, USA (Jul 2022). <https://doi.org/10.1145/3477495.3531757>
 16. Su, Z., Wang, D., Miao, C.: Context Pooling: Query-specific Graph Pooling for Generic Inductive Link Prediction in Knowledge Graphs. In: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2. pp. 2680–2689. KDD ’25, Association for Computing Machinery, New York, NY, USA (Aug 2025). <https://doi.org/10.1145/3711896.3736890>

17. Geng, Y., Chen, J., Pan, J.Z., Chen, M., Jiang, S., Zhang, W., Chen, H.: Relational Message Passing for Fully Inductive Knowledge Graph Completion. In: 2023 IEEE 39th International Conference on Data Engineering (ICDE). pp. 1221–1233 (Apr 2023). <https://doi.org/10.1109/ICDE55515.2023.00098>
18. Lee, J., Chung, C., Whang, J.J.: INGRAM: Inductive knowledge graph embedding via relation graphs. In: Proceedings of the 40th International Conference on Machine Learning. ICML’23, vol. 202, pp. 18796–18809. JMLR.org, Honolulu, Hawaii, USA (Jul 2023)
19. Ding, L., Huang, L., Yu, Z., Jin, D., He, D.: Towards Global-Topology Relation Graph for Inductive Knowledge Graph Completion. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 11581–11589 (Apr 2025). <https://doi.org/10.1609/aaai.v39i11.33260>
20. Galkin, M., Yuan, X., Mostafa, H., Tang, J., Zhu, Z.: Towards Foundation Models for Knowledge Graph Reasoning. International Conference on Learning Representations **2024**, 31598–31619 (May 2024)
21. Du, E., Liu, S., Zhang, Y.: GraphOracle: Efficient Fully-Inductive Knowledge Graph Reasoning via Relation-Dependency Graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 40, pp. 19055–19063 (Mar 2026). <https://doi.org/10.1609/aaai.v40i23.38978>
22. Yang, F., Yang, Z., Cohen, W.W.: Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In: Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017)
23. Sadeghian, A., Armandpour, M., Ding, P., Wang, D.Z.: DRUM: End-To-End Differentiable Rule Mining On Knowledge Graphs (Oct 2019). <https://doi.org/10.48550/arXiv.1911.00055>
24. Qu, M., Chen, J., Xhonneux, L.P., Bengio, Y., Tang, J.: RNNLogic: Learning Logic Rules for Reasoning on Knowledge Graphs. In: International Conference on Learning Representations (2021)
25. Zhu, Z., Zhang, Z., Xhonneux, L.P., Tang, J.: Neural bellman-ford networks: A general graph neural network framework for link prediction. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. pp. 29476–29490. NIPS ’21, Curran Associates Inc., Red Hook, NY, USA (Dec 2021)
26. Zhang, Y., Yao, Q.: Knowledge Graph Reasoning with Relational Digraph. In: Proceedings of the ACM Web Conference 2022. pp. 912–924. WWW ’22, Association for Computing Machinery, New York, NY, USA (Apr 2022). <https://doi.org/10.1145/3485447.3512008>
27. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2D knowledge graph embeddings. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence. pp. 1811–1818. AAAI’18/IAAI’18/EAAI’18, AAAI Press, New Orleans, Louisiana, USA (Feb 2018)
28. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Allauzen, A., Grefenstette, E., Hermann, K.M., Larochelle, H., Yih, S.W.t. (eds.) Proceedings of the 3rd Workshop on Continuous Vector Space Models and Their Compositionality. pp. 57–66. Association for Computational Linguistics, Beijing, China (Jul 2015). <https://doi.org/10.18653/v1/W15-4007>
29. Feng, Y., You, H., Zhang, Z., Ji, R., Gao, Y.: Hypergraph Neural Networks (Feb 2019). <https://doi.org/10.48550/arXiv.1809.09401>

30. Galkin, M., Trivedi, P., Maheshwari, G., Usbeck, R., Lehmann, J.: Message Passing for Hyper-Relational Knowledge Graphs. In: Webber, B., Cohn, T., He, Y., Liu, Y. (eds.) *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pp. 7346–7359. Association for Computational Linguistics, Online (Nov 2020). <https://doi.org/10.18653/v1/2020.emnlp-main.596>
31. Yang, Y., Li, X., Guan, Y., Wang, H., Kong, C., Jiang, J.: LHP: Logical hypergraph link prediction. *Expert Systems with Applications* **222**, 119842 (Jul 2023). <https://doi.org/10.1016/j.eswa.2023.119842>
32. Fatemi, B., Taslakian, P., Vazquez, D., Poole, D.: Knowledge Hypergraph Embedding Meets Relational Algebra (Feb 2021). <https://doi.org/10.48550/arXiv.2102.09557>
33. Xiong, W., Hoang, T., Wang, W.Y.: DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In: Palmer, M., Hwa, R., Riedel, S. (eds.) *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. pp. 564–573. Association for Computational Linguistics, Copenhagen, Denmark (Sep 2017). <https://doi.org/10.18653/v1/D17-1060>
34. Gesese, G.A., <https://orcid.org/0000-0003-3807-7145>, View Profile, Sack, H., <https://orcid.org/0000-0001-7069-9804>, View Profile, Alam, M., <https://orcid.org/0000-0002-7867-6612>, View Profile: RAILD: Towards Leveraging Relation Features for Inductive Link Prediction In Knowledge Graphs. In: *Proceedings of the 11th International Joint Conference on Knowledge Graphs*. pp. 82–90. ACM Other Conferences (Oct 2022). <https://doi.org/10.1145/3579051.3579066>
35. Meilicke, C., Fink, M., Wang, Y., Ruffinelli, D., Gemulla, R., Stuckenschmidt, H.: Fine-Grained Evaluation of Rule- and Embedding-Based Systems for Knowledge Graph Completion. In: Vrandečić, D., Bontcheva, K., Suárez-Figueroa, M.C., Presutti, V., Celino, I., Sabou, M., Kaffee, L.A., Simperl, E. (eds.) *The Semantic Web – ISWC 2018*. pp. 3–20. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-030-00671-6_1
36. Mai, S., Zheng, S., Yang, Y., Hu, H.: Communicative Message Passing for Inductive Relation Reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence* **35**(5), 4294–4302 (May 2021). <https://doi.org/10.1609/aaai.v35i5.16554>
37. Xu, X., Zhang, P., He, Y., Chao, C., Yan, C.: Subgraph Neighboring Relations Infomax for Inductive Link Prediction on Knowledge Graphs. In: *Thirty-First International Joint Conference on Artificial Intelligence*. vol. 3, pp. 2341–2347 (Jul 2022). <https://doi.org/10.24963/ijcai.2022/325>
38. Liu, S., Grau, B., Horrocks, I., Kostylev, E.: INDIGO: GNN-Based Inductive Knowledge Graph Completion Using Pair-Wise Encoding. In: *Advances in Neural Information Processing Systems*. vol. 34, pp. 2034–2045. Curran Associates, Inc. (2021)
39. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.: Composition-based Multi-Relational Graph Convolutional Networks. In: *Eighth International Conference on Learning Representations* (Apr 2020)
40. Galkin, M., Denis, E., Wu, J., Hamilton, W.L.: NodePiece: Compositional and Parameter-Efficient Representations of Large Knowledge Graphs. In: *International Conference on Learning Representations* (Oct 2021)
41. Daza, D., Cochez, M., Groth, P.: Inductive Entity Representations from Text via Link Prediction. In: *Proceedings of the Web Conference 2021*. pp. 798–808. WWW '21, Association for Computing Machinery, New York, NY, USA (Jun 2021). <https://doi.org/10.1145/3442381.3450141>

42. Ali, M., Berrendorf, M., Galkin, M., Thost, V., Ma, T., Tresp, V., Lehmann, J.: Improving Inductive Link Prediction Using Hyper-relational Facts. In: Hotho, A., Blomqvist, E., Dietze, S., Fokoue, A., Ding, Y., Barnaghi, P., Haller, A., Dragoni, M., Alani, H. (eds.) *The Semantic Web – ISWC 2021*. pp. 74–92. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-88361-4_5