FAITHFUL RULE LEARNING FOR TABULAR CELL COMPLETION

Anonymous authors
Paper under double-blind review

ABSTRACT

Tabular cell completion aims to infer values that can fill missing cells in incomplete table rows. While machine learning models have achieved strong performance on this task, their limited interpretability restricts their applicability in high-stakes domains. In this paper, we introduce two interpretable models for tabular cell completion based on path-based relational patterns. Given an incomplete row and a candidate value, the models aggregate evidence from paths connecting the known constants in the row to the candidate value. A key feature of our approach is that the learned models admit *faithful* symbolic characterizations in the form of equivalent Datalog programs: for every model instance, we show how to extract an equivalent Datalog program that produces exactly the same outputs on every database over the underlying schema. We further analyze the relationship between the two models, showing how different aggregation mechanisms lead to distinct trade-offs between expressive power and simplicity of rule extraction. Experimental results demonstrate strong performance on tabular cell completion benchmarks while additionally providing human-readable explanations with formal faithfulness guarantees.

1 INTRODUCTION

Tabular data underlies a wide range of real-world applications, from medical records to financial transactions. In practice, tabular data is often incomplete: missing entries arise due to collection errors, privacy restrictions, or unavailable information. The task of *tabular cell completion* aims to infer plausible values for these missing entries based on the observed data (Zhang & Balog, 2019; Ahmadov et al., 2015; Yakout et al., 2012). Unlike classical imputation methods in machine learning, which typically produce a single best estimate under a fixed statistical model (Sun et al., 2023; Van Buuren & Groothuis-Oudshoorn, 2011; Stekhoven & Bühlmann, 2012), tabular cell completion allows for multiple valid candidates, reflecting the inherent ambiguity of real-world data.

Early approaches to this problem relied on statistical heuristics (Yakout et al., 2012), external resources such as knowledge bases (Zhang & Balog, 2019), or specialized settings restricted to particular table components (e.g., headers) (Ahmadov et al., 2015). More recently, machine learning methods have modeled tabular data as *hypergraphs*, where nodes represent cell values and hyperedges correspond to rows, and have applied hypergraph neural networks to perform completion (Yadati et al., 2019; Yadati, 2020; Huang et al., 2025). While effective, these models typically lack transparent and provably faithful explanations for their outputs. This limitation reduces trust in sensitive application domains, such as medical record completion, legal compliance, and financial data reconciliation, where auditability and interpretability are critical for adoption.

Our task is related to neural rule learning methods for knowledge graph completion, such as Neural-LP (Yang et al., 2017) and DRUM (Sadeghian et al., 2019). However, these methods are primarily designed for binary relations, as is standard in knowledge graphs. In contrast, tabular data involves higher-arity relations, which introduce additional modeling and reasoning challenges. Taken together, existing approaches either lack faithful interpretability or are limited to binary relations. As a result, no existing approach provides formally grounded and faithful rule-based explanations for tabular data completion over higher-arity relations.

We address this challenge by drawing on ideas from faithful rule learning (Tena Cucala et al., 2022a;b; 2023; Wang et al., 2024a). Unlike feature-attribution methods for explaining tabular mod-

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.

ID	Drug	TargetGene
1	*Aspirin*	*COX-1*
2	*Ibuprofen*	*COX-1*
...	...	...

Gene	Pathway	Disease
COX-1	*Prostaglandin-synthesis*	*Colorectal-cancer*
...	...	...

Drug	SharedTargetGene	Drug
Aspirin	*COX-2*	*Celecoxib*
...	...	...

Drug	PreventDisease
Celecoxib	*Colorectal-cancer*
...	...
Aspirin	**?**

Figure 1: An example of tabular cell completion over multi-source biomedical tables relating drugs, targets, and diseases. The symbol '?' denotes a missing cell value.

els (Lundberg & Lee, 2017), our goal is to design models for tabular cell completion that are intrinsically interpretable. Crucially, we provide a formal guarantee of faithfulness: every output produced by the model can be derived from the extracted Datalog rules on the same input database, and conversely, the rules derive exactly the outputs of the model. For example, as illustrated in Figure 1, a prediction that a missing value is *Colorectal-cancer* can be justified by a rule relating shared drug targets and disease pathways. Such rules provide a transparent and verifiable account of the model's reasoning process, enabling practitioners to inspect, verify, and reuse learned patterns for tasks such as data cleaning, auditing, and integration.

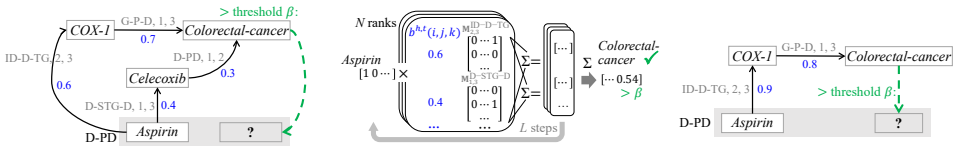
Our contributions are as follows: (1) Two interpretable models for tabular cell completion based on sum and max aggregation over relational paths (Sections 3 and 5); (2) Faithful rule extraction methods for both models, providing exact symbolic characterizations with formal guarantees (Sections 4 and 5); notably, the max-based model enables simpler and more efficient rule extraction; and (3) A comprehensive empirical evaluation demonstrating competitive performance against strong baselines together with the practical feasibility of rule extraction (Section 6).

2 Background

Tabular Cell Completion. We adopt standard database terminology. A schema consists of two disjoint sets of *predicates*, corresponding to tables, and *constants*, corresponding to values. Each predicate P has an *arity* $|P|$ (its number of columns). A *fact* is an expression $P(c_1, \ldots, c_{|P|})$, where P is a predicate and each c_i is either a constant or the special *null* symbol '?'. Intuitively, a fact corresponds to a row in a table, and '?' denotes a missing value. A *table* over predicate P is a finite set of facts over P. A database (instance) is a collection of such tables. A fact, table, or database is *incomplete* if it contains at least one null value. For simplicity, we focus on the case where each fact contains at most one null value; extensions to multiple null values are discussed in the appendix. For example, the table in Figure 1 with columns 'Drug' and 'PreventDisease' is represented as a binary predicate `D-PD`, and the incomplete row (Drug = Aspirin, PreventDisease = ?) corresponds to the fact `D-PD`(*Aspirin*, ?). For an incomplete database $\mathcal{D}$, a *completion* is obtained by replacing each null with exactly one constant occurring in $\mathcal{D}$. For example, in Figure 1, a missing value '?' may be replaced with '*Colorectal-cancer*'. The task of *tabular cell completion* is therefore to map each incomplete fact in $\mathcal{D}$ to the set of constants that may replace its null value (Yakout et al., 2012).

This task is related to *hypergraph link prediction* (Chen & Liu, 2023; Fatemi et al., 2020; Yadati et al., 2019; Wang et al., 2023a), as a database can be viewed as a hypergraph with constants as nodes and facts as hyperedges. However, unlike link prediction that predicts arbitrary facts, our task specifically fills missing cells in partially observed rows. While hypergraph methods can be adapted by discarding incomplete facts, converting the remaining data into a hypergraph, applying a model, and extracting outputs, they lack direct mechanisms for reasoning over incompleteness. The task is also related to *Programming by Example (PBE)* (Gulwani, 2011; Wang et al., 2017; Kolb et al., 2017). In PBE users provide input-output pairs and derivation knowledge, whereas in our setting completions are inferred automatically from the data (see Appendix A for additional related work).

Datalog. Datalog is a rule-based language widely used in databases, logic programming, and knowledge representation (Abiteboul et al., 1995). A *term* is either a variable or a constant. An *atom* has the form $P(t_1, \ldots, t_{|P|})$, where P is a predicate and each t_i is a term. An *inequality* is an expression of the form $t_1 \not\approx t_2$, where t_1 and t_2 are terms. A *literal* is either an atom or an inequality. A *fact* is a variable-free atom, and a *dataset* is a finite set of facts. A (Datalog) *rule* is an expression of the form $H \leftarrow B_1 \wedge \ldots \wedge B_\ell$, where $\ell \geq 0$, H is the *head atom*, and each B_i is a *body literal*. A (Datalog) *program* is a finite set of such rules.



(a) MC: aggregate all paths of length ≤ L (b) MC: computation process (c) MC-max: keep the max-weighted path

Figure 2: An illustration of our approach: the model aggregates paths from known constants in the incomplete fact (e.g., *Aspirin*) to a candidate constant (e.g., *Colorectal-cancer*). Each path contributes a weighted signal, and the model outputs the candidate if the aggregated score exceeds a threshold β. The values in blue are learnable parameters.

For a mapping σ from variables to terms, and for ω a term, literal, or conjunction of literals, we write $\omega\sigma$ for the result of replacing each variable x in ω with $\sigma(x)$ (when defined). Two conjunctions of literals ω_1 and ω_2 are *isomorphic* if there exists a bijection σ from the variables in ω_1 to those in ω_2 such that $\omega_1\sigma = \omega_2$. A set S contains a conjunction of literals ω_1 *up to isomorphism* if there exists $\omega_2 \in S$ such that ω_1 and ω_2 are isomorphic. Intuitively, isomorphism allows us to treat rule bodies as equivalent if they have the same structure up to variable renaming.

Each rule ϱ defines an *immediate consequence* operator T_ϱ. For a dataset $\mathcal{D}$, $T_\varrho(\mathcal{D})$ is the dataset containing all facts of the form $H\sigma$, where σ is a mapping from the variables in ϱ to constants in $\mathcal{D}$ such that each body literal B_i is satisfied by σ. Specifically, if B_i is an atom, then $B_i\sigma \in \mathcal{D}$, and if B_i is an inequality of the form $x \not\approx y$, then $x\sigma \neq y\sigma$, for all $1 \leq i \leq \ell$. Intuitively, $T_\varrho(\mathcal{D})$ derives all facts that can be inferred by applying rule ϱ to $\mathcal{D}$. Inequalities are interpreted under the standard *Unique Name Assumption (UNA)*, i.e., different constants refer to different objects. Following Dantsin et al. (2001); Wang et al. (2024a), we use Datalog extended so that a rule body may contain finite disjunctions of inequalities; such a disjunction is satisfied by a grounding σ if and only if at least one of the disjuncts is satisfied by σ under the UNA. It is only a compact notation since distributing body disjunctions yields an equivalent finite disjunction-free Datalog program. The immediate consequence operator for a program $\mathcal{R}$ is defined as $T_\mathcal{R}(\mathcal{D}) = \bigcup_{\varrho\in\mathcal{R}} T_\varrho(\mathcal{D})$. Since each dataset contains finitely many constants, only finitely many facts can be derived.

Tensors, Vectors, Matrices. We consider n-dimensional *tensors* over $\mathbb{R}$. A *vector* is a 1-dimensional tensor, and a *matrix* is a 2-dimensional tensor. For a tensor $\mathbf{A}$, we write $\mathbf{A}(i_1, i_2, \ldots, i_n)$ for its element at position $(i_1, i_2, \ldots, i_n)$. For tensors $\mathbf{T}_1, \ldots, \mathbf{T}_n$ of the same dimension, $\max_{1\leq i\leq n} \mathbf{T}_i$ denotes their element-wise maximum. For matrices $\mathbf{M} \in \mathbb{R}^{m\times n}$ and $\mathbf{N} \in \mathbb{R}^{n\times p}$, the *max-product* $\mathbf{M} \otimes \mathbf{N}$ is the matrix in $\mathbb{R}^{m\times p}$ defined by $(\mathbf{M} \otimes \mathbf{N})(i, j) = \max_{1\leq k\leq n} \mathbf{M}(i, k) \cdot \mathbf{N}(k, j)$ for $1 \leq i \leq m$ and $1 \leq j \leq p$.

3 A Model for Tabular Cell Completion

This section introduces our model and provides a formal interpretation of its computation. We assume an arbitrary database schema $P_1, \ldots, P_\delta$ throughout the paper. Figure 2 illustrates the key intuition. Given an incomplete fact, the model considers paths in the database that connect the known constants in the fact to a candidate constant. Each path contributes a weighted signal, and the model aggregates these contributions to determine which candidates can fill the missing value.

3.1 Model Definition

Our method builds on rule-based link prediction models for graphs, which infer relationships by aggregating evidence from paths between constants (Yang et al., 2017; Sadeghian et al., 2019; Tena Cucala et al., 2022b; Wang et al., 2024a). We extend this idea to relational tables of arbitrary arity, where paths capture join-like connections between constants across facts.

Definition 1. *A* path *of length $\ell \geq 1$ in a database $\mathcal{D}$ is a sequence of constants $c_0, \ldots, c_\ell$, possibly with repetitions, such that each consecutive pair occurs in distinct positions of a complete fact in $\mathcal{D}$. A* path schema *of length ℓ is a sequence $(P_{r_1}, p_1, q_1), \ldots, (P_{r_\ell}, p_\ell, q_\ell)$, where $1 \leq r_i \leq \delta$ and $1 \leq p_i, q_i \leq |P_{r_i}|$ with $p_i \neq q_i$. The path* matches *the schema if, for every $1 \leq i \leq \ell$, some complete fact over P_{r_i} in $\mathcal{D}$ contains c_{i-1} and c_i at positions p_i and q_i, respectively.*

Example 1. *In Figure 1, the facts* $\texttt{ID-D-TG}(1, \textit{Aspirin}, \textit{COX-1})$ *and* $\texttt{G-P-D}(\textit{COX-1}, \textit{Prostaglandin-synthesis}, \textit{Colorectal-cancer})$ *give the path* $\textit{Aspirin}, \textit{COX-1}, \textit{Colorectal-cancer}$*, matching the schema* $(\texttt{ID-D-TG}, 2, 3), (\texttt{G-P-D}, 1, 3)$*. Predicate names abbreviate column names. A path may match multiple schemas.*

Path schemas abstract paths by ignoring specific constants and retaining only the sequence of predicates and positions used to traverse the database. Ideally, one could assign a learnable parameter to each path schema, reflecting its contribution for a given predicate. However, such a parameterization would require an extremely large number of parameters, most of which would rarely be used in practice. Following TensorLog (Cohen et al., 2020), Neural-LP (Yang et al., 2017), and DRUM (Sadeghian et al., 2019), we instead share weights across path schemas by assigning them to individual predicate-position transitions and multiplying them along each path realization. For example, the two transitions through *COX-1* in Figure 2(a) contribute $0.6 \times 0.7 = 0.42$.

Definition 2. *A* Multi-Chain (MC) model *of rank* $N \geq 1$ *and depth* $L \geq 1$ *is a tuple*

$$(\mathbf{b}^{1,1}, \ldots, \mathbf{b}^{1,|P_1|}, \ldots, \mathbf{b}^{\delta,1}, \ldots, \mathbf{b}^{\delta,|P_\delta|}, \beta)\,,$$

where $\beta \in \mathbb{R}_{\geq 0}$ *and each* $\mathbf{b}^{h,t}$ *is a tensor in* $[0,1]^{N \times L \times K}$ *with* $K = \sum_{k=1}^{\delta}(|P_k| \cdot (|P_k| - 1)) + 1$.

The depth L bounds the path length, while the rank N provides multiple sets of transition weights whose contributions are aggregated. For predicting position t of predicate P_h, $\mathbf{b}^{h,t}(i,j,k)$ assigns a weight to transition k at step j in rank component i. Indices $1, \ldots, K-1$ enumerate all triples (P, p, q) with $p \neq q$; index K denotes the no-transition symbol $\top$, allowing paths shorter than L. Figure 2(b) illustrates the computation.

Let $\mathcal{M}$ be an MC model and $P_h(\mathbf{c})$ an incomplete fact in $\mathcal{D}$ with a null at position t. Enumerate the distinct constants in $\mathcal{D}$ as $c_1, \ldots, c_\epsilon$. For each predicate P_k and distinct positions p, q, define $\mathbf{M}^k_{p,q} \in \{0,1\}^{\epsilon \times \epsilon}$ by $\mathbf{M}^k_{p,q}(s,u) = 1$ exactly when some complete fact over P_k in $\mathcal{D}$ contains c_s, c_u at positions p, q, respectively.

For each rank component $1 \leq i \leq N$ and known constant c_s in $\mathbf{c}$, initialize $\mathbf{v}^{i,0}_{c_s}$ as the one-hot vector with its s-th entry equal to 1. For $j = 1, \ldots, L$, the following update accumulates weighted contributions of paths of length at most j from c_s to each candidate:

$$(\mathbf{v}^{i,j}_{c_s})^\intercal = \mathbf{b}^{h,t}(i,j,K)\,(\mathbf{v}^{i,j-1}_{c_s})^\intercal + (\mathbf{v}^{i,j-1}_{c_s})^\intercal \left(\sum_{k=1}^{\delta} \sum_{\substack{1 \leq p,q \leq |P_k| \\ p \neq q}} \mathbf{b}^{h,t}(i,j,\mathtt{d}_{k,p,q}) \cdot \mathbf{M}^k_{p,q} \right), \tag{1}$$

where $\mathtt{d}_{k,p,q}$ indexes the triple (k,p,q). The first term leaves paths unchanged, while the second extends them by one relational transition. The final score vector is $\mathbf{v}^{h,t}_{\mathbf{c}} = \sum_{i=1}^{N} \sum_{c_s \in \mathbf{c}} \mathbf{v}^{i,L}_{c_s}$, where the inner sum ranges over known constant positions. The model outputs c_u exactly when $\mathbf{v}^{h,t}_{\mathbf{c}}(u) > \beta$. We write $T_{\mathcal{M}}(\mathcal{D})$ for the set of completed facts produced by these predictions.

3.2 Model Interpretation

Let Ω contain all path schemas of length at most L and the no-transition $\top$. The following lemma decomposes the MC score into parameter-dependent schema weights and data-dependent path counts.

Lemma 1. *Let* $\mathcal{M}$ *be an MC model with rank* N *and depth* L *as in Definition 2. Let* $P_h(\mathbf{c})$ *be an incomplete fact in* $\mathcal{D}$ *with a null value in position* $1 \leq t \leq |P_h|$*. Then,* $\mathbf{v}^{h,t}_{\mathbf{c}}$ *can be written as*

$$\mathbf{v}^{h,t}_{\mathbf{c}} = \sum_{\omega \in \Omega} \mathtt{wt}^{h,t}_{\mathcal{M}}(\omega) \cdot \mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}\,,$$

where $\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}$ *is a non-negative vector dependent only on the data (and independent of the model parameters) defined as follows: (1) if* $\omega = \top$*, then* $\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}(u)$ *counts the occurrences of* c_u *in* $\mathbf{c}$*; (2) if* ω *is a path schema of length* $\ell \geq 1$*, then* $\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}(u)$ *counts the distinct paths in* $\mathcal{D}$ *matching* ω *and connecting constants in* $\mathbf{c}$ *to* c_u*. Moreover, the weight* $\mathtt{wt}^{h,t}_{\mathcal{M}}(\omega)$ *is given by*

$$\mathtt{wt}^{h,t}_{\mathcal{M}}(\omega) = \sum_{(d_1,\ldots,d_L) \in S^L_\omega} \sum_{i=1}^{N} \prod_{j=1}^{L} \mathbf{b}^{h,t}(i,j,d_j)\,, \tag{2}$$

where $S^L_\omega \subseteq \{1,\dots,K\}^L$ is the set of all sequences $(d_1,\dots,d_L)$ obtained by placing the steps of ω in order into L positions and filling the remaining positions with the symbol K (corresponding to $\top$). In particular, for $\omega = \top$, we have $S^L_\top = \{(K,\dots,K)\}$.

Since all schema weights are nonnegative, the score is monotonic in the path counts. Once a schema's contribution exceeds β, additional matching paths cannot change the prediction. For each predicate P_h and position t, we define the *maximum cardinality* $\mathring{C}^{h,t} : \Omega \to \mathbb{Z}_{\geq 0}$ as

$$\mathring{C}^{h,t}(\omega) = \begin{cases} 0 & \text{if } \mathtt{wt}^{h,t}_{\mathcal{M}}(\omega) = 0\,, \\ \left\lfloor \beta / \mathtt{wt}^{h,t}_{\mathcal{M}}(\omega) \right\rfloor + 1 & \text{otherwise.} \end{cases} \tag{3}$$

Zero-weight path schemas can be ignored. For every positive-weight schema, $\mathring{C}^{h,t}(\omega)\,\mathtt{wt}^{h,t}_{\mathcal{M}}(\omega) > \beta$. Thus, truncating each schema's path count at this bound preserves the threshold decision: either no positive contribution is changed, or a truncated contribution alone still exceeds β. This permits faithful rule extraction using finitely many copies of each schema (Section 4).

4 Faithful Rule Extraction for the MC Model

In this section, we show how to extract Datalog rules from an MC model. These rules are *faithful*, meaning that they produce exactly the same outputs as the model on any database. We also describe a simpler variant that extracts rules explaining individual model outputs.

4.1 Multichain Rules

To apply Datalog rules, we represent each database over the fixed database schema $\{P_i\}_{1\leq i\leq \delta}$ as a relational dataset. We encode an incomplete fact $P_i(c_1,\dots,c_{t-1},?,c_{t+1},\dots,c_{|P_i|})$ as $P^t_i(c_1,\dots,c_{t-1},c_{t+1},\dots,c_{|P_i|})$, where the auxiliary predicate P^t_i has arity $|P_i|-1$ and records the missing position t. For example, $\texttt{D-PD}(\textit{Aspirin},?)$ is encoded as $\texttt{D-PD}^2(\textit{Aspirin})$.

We next introduce *chain patterns*, which represent path schemas as sequences of relational joins.

Definition 3. *A* chain pattern $\Phi(x,y)$ *of length $\ell \geq 1$ is an ordered conjunction of ℓ atoms.*

For each $1 \leq i \leq \ell$, the i-th atom contains two distinguished linking variables z_{i-1} *and z_i, each occurring exactly once, with $z_0 = x$ and $z_\ell = y$, while all other variables occur at most once.*

A path schema is translated into a chain pattern by placing z_{i-1} and z_i at positions p_i and q_i of each atom P_{r_i} and filling the remaining positions with fresh variables. For example, $(\texttt{ID-D-TG},2,3),(\texttt{G-P-D},1,3)$ corresponds to $\Phi(x,y) = \texttt{ID-D-TG}(v,x,z) \wedge \texttt{G-P-D}(z,w,y)$. To express the bounded path counts needed for MC rule extraction, we combine copies of a chain pattern with inequalities requiring distinct paths.

Definition 4. *A* multichain pattern *for variables x and y with* core *chain pattern $\Phi(x,y)$ and cardinality $C \in \mathbb{N}$ is of the form*

$$\bigwedge_{j=1}^{C} \Phi^j(x,y) \;\wedge \bigwedge_{1\leq j<j'\leq C} \left(\bigvee_{i=1}^{\ell-1} z_i^j \not\approx z_i^{j'} \right),$$

where $\Phi^j(x,y)$ is obtained from $\Phi(x,y)$ by replacing each internal linking variable z_i ($1 \leq i \leq \ell-1$) with a fresh variable z_i^j, and all other variables except x and y are chosen fresh.

The inequalities require each pair of copies to differ at an internal linking variable, ensuring at least C distinct paths between the endpoints.

Example 2. *A multichain pattern with the above chain pattern $\Phi(x,y)$ and cardinality 2 is*

$$\textit{ID-D-TG}(v_1,x,z_1) \wedge \textit{G-P-D}(z_1,w_1,y) \wedge \textit{ID-D-TG}(v_2,x,z_2) \wedge \textit{G-P-D}(z_2,w_2,y) \wedge z_1 \not\approx z_2\,.$$

The two paths must therefore pass through different intermediate constants assigned to z_1 and z_2.

An MC rule combines an incomplete-fact atom with multichain patterns connecting its known positions to the predicted value.

Definition 5. *A* Multichain (MC) rule *is of the form* (4).

$$P_h(x'_1,\dots,x'_{t-1},y,x'_{t+1},\dots,x'_{|P_h|}) \leftarrow P_h^t(x'_1,\dots,x'_{t-1},x'_{t+1},\dots,x'_{|P_h|}) \;\wedge \bigwedge_{\substack{1\le r\le |P_h|\\ r\neq t}} \varphi_r(x'_r,y)\,. \quad (4)$$

For each $1 \le r \le |P_h|$ with $r \neq t$, the variable x'_r is either x_r or equal to y, and $\varphi_r(x'_r, y)$ is a (possibly empty) conjunction of multichain patterns over x'_r and y, sharing no variables other than x'_r and y. Moreover, if none of the variables x'_r is equal to y, then at least one of the multichain patterns $\varphi_r(x'_r, y)$ must be non-empty.

Intuitively, an MC rule outputs a value by requiring that enough distinct paths, matching specific patterns, connect the known constants in an incomplete fact to the output value.

4.2 Faithful Rule Extraction for an MC Model

A program $\mathcal{R}$ is *faithful* to $\mathcal{M}$ if $T_{\mathcal{R}}(\mathcal{D}) = T_{\mathcal{M}}(\mathcal{D})$ for every database $\mathcal{D}$, by interpreting the derived facts as completed rows (Tena Cucala et al., 2022b; Wang et al., 2024a).

The cardinality bounds in Equation (3) allow us to enumerate finitely many candidate rules and retain those whose required path evidence exceeds the extraction threshold.

Definition 6. *Fix a predicate P_h and target position t. Let $\Theta^{h,t}$ contain all conjunctions $\varphi(x, y)$ obtained by choosing, for each $\omega \in \Omega \setminus \{\top\}$, a multichain pattern with the corresponding core and cardinality $C_\omega(\varphi) \in \{0,\dots,\mathring{C}^{h,t}(\omega)\}$, where cardinality zero means that the pattern is omitted. Include also each variant $\varphi\{x \mapsto y\}$. Define their scores by*

$$\texttt{score}^{h,t}_{\mathcal{M}}(\varphi) = \sum_{\omega\in\Omega\setminus\{\top\}} C_\omega(\varphi)\,\texttt{wt}^{h,t}_{\mathcal{M}}(\omega)\,,\ \texttt{score}^{h,t}_{\mathcal{M}}(\varphi\{x\mapsto y\}) = \texttt{score}^{h,t}_{\mathcal{M}}(\varphi) + \texttt{wt}^{h,t}_{\mathcal{M}}(\top)\,.$$

For an MC rule ϱ of the form (4) *with $\varphi_r \in \Theta^{h,t}$ for each $r \neq t$, let $\texttt{score}^{h,t}_{\mathcal{M}}(\varrho) = \sum_{r\neq t} \texttt{score}^{h,t}_{\mathcal{M}}(\varphi_r)$. For an extraction threshold γ, let $\mathcal{R}^{h,t}_{\mathcal{M}}$ contain all such rules with $\texttt{score}^{h,t}_{\mathcal{M}}(\varrho) > \gamma$, and set $\mathcal{R}_{\mathcal{M}} = \bigcup_{h,t} \mathcal{R}^{h,t}_{\mathcal{M}}$.*

The following theorem establishes the correctness of this construction and an upper bound on the number of extracted rules.

Theorem 1. *The program $\mathcal{R}_{\mathcal{M}}$ is faithful to $\mathcal{M}$ when $\gamma = \beta$, with $|\mathcal{R}_{\mathcal{M}}| \le \delta\nu\big[2(C+1)^{|\Omega|-1}\big]^{\nu-1}$, where $\nu = \max_{1\le k\le\delta} |P_k|$ and $C = \max_{h,t} \max_{\omega\in\Omega,\,\omega\neq\top} \mathring{C}^{h,t}(\omega)$.*

4.3 Explaining Database-specific Outputs

To explain predictions on a fixed database $\mathcal{D}$, we construct an MC rule for each predicted completion using the observed path counts, truncated at $\mathring{C}^{h,t}(\omega)$. Zero-cardinality patterns are omitted, and source variables are identified with the target variable whenever their constants coincide. This yields a database-specific program $\mathcal{R}_{\mathcal{M},\mathcal{D}}$ without enumerating all combinations of cardinalities. The formal construction is given in Appendix B.3.

Theorem 2. *When $\gamma = \beta$, the program $\mathcal{R}_{\mathcal{M},\mathcal{D}}$ satisfies $\mathcal{R}_{\mathcal{M},\mathcal{D}} \subseteq \mathcal{R}_{\mathcal{M}}$ and $T_{\mathcal{R}_{\mathcal{M},\mathcal{D}}}(\mathcal{D}) = T_{\mathcal{M}}(\mathcal{D})$. Moreover, the program $\mathcal{R}_{\mathcal{M},\mathcal{D}}$ can be constructed in time $\mathcal{O}\big(\epsilon^{L+\nu}\big)$, where $\nu = \max_{1\le k\le\delta} |P_k|$, and ϵ is the number of constants in $\mathcal{D}$.*

The extracted rules are sound on every database, but completeness is guaranteed only on $\mathcal{D}$.

5 MC-max: A Model with Efficient Rule Extraction

Recall that the MC model predicts that a constant c completes an incomplete fact by aggregating contributions from all paths in the database connecting the known constants in the fact to c. In practice, however, aggregating all paths can make the model sensitive to less informative paths with small weights. To address this, MC-max replaces the accumulation of path evidence with selection of the highest-weight path realization (Figure 2(c)). This simplification limits expressive power (Appendix D) but enables substantially more efficient rule extraction with lower complexity.

5.1 Model Definition

Definition 7. *An* MC-max model *has the same parameters as an MC model (Definition 2), but replaces additive aggregation with max aggregation. The vectors $\mathbf{v}_{c_s}^{i,j}$ are computed as follows:*

$$(\mathbf{v}_{c_s}^{i,j})^\intercal = \max\Big\{\, \mathbf{b}^{h,t}(i,j,K)\cdot(\mathbf{v}_{c_s}^{i,j-1})^\intercal,\ \max_{\substack{1\le k\le \delta\\ 1\le p,q\le |P_k|,\ p\neq q}} \mathbf{b}^{h,t}(i,j,\mathtt{d}_{k,p,q})\cdot(\mathbf{v}_{c_s}^{i,j-1})^\intercal \otimes \mathbf{M}_{p,q}^{k}\Big\}\,, \tag{5}$$

where $\otimes$ denotes the max-product operation. The final output vector for predicate P_h and position t is $\mathbf{v}_{\mathbf{c}}^{h,t} = \max_{1\le i\le N,\ c_s\in\mathbf{c}} \mathbf{v}_{c_s}^{i,L}$.

The initialization and prediction of MC-max follow MC: $\mathbf{v}_{c_s}^{i,0}$ is one-hot, and c_u is output exactly when $\mathbf{v}_{\mathbf{c}}^{h,t}(u) > \beta$. MC-max can therefore be viewed as a tractable approximation of MC that replaces bounded counting by path existence reasoning.

5.2 Faithful Rule Extraction for MC-max

For MC-max, a schema's weight is the maximum over its realizations and rank components, replacing the sums in Equation (2).

Definition 8. *For each predicate P_h and position t, the* MC-max weight *of a path schema ω relative to the model $\mathcal{M}$ is defined by*

$$\mathtt{wtm}_{\mathcal{M}}^{h,t}(\omega) = \max_{\substack{1\le i\le N\\ (d_1,\dots,d_L)\in S_\omega^L}} \prod_{j=1}^{L} \mathbf{b}^{h,t}(i,j,d_j)\,. \tag{6}$$

Definition 9. *An* MC-max rule *is a rule of the form*

$$P_h(x_1,\dots,x_{t-1},y,x_{t+1},\dots,x_{|P_h|}) \leftarrow P_h^t(x_1,\dots,x_{t-1},x_{t+1},\dots,x_{|P_h|}) \wedge \Phi(x_r,y)\,, \tag{7}$$

where $1\le t\le |P_h|$ is the predicted position, $r\neq t$, and $\Phi(x_r,y)$ is the chain pattern corresponding to a path schema ω (Definition 3). If $\omega=\top$, then Φ is empty and $x_r = y$.

For a threshold γ, the MC-max program $\mathcal{R}_{\mathcal{M}}^{\max}$ *is the set of all MC-max rules corresponding to path schemas ω satisfying $\mathtt{wtm}_{\mathcal{M}}^{h,t}(\omega) > \gamma$.*

Each rule tests the existence of a single path schema whose weight exceeds γ, while no path-count constraints or inequalities are needed.

Theorem 3. *Let $\mathcal{M}$ be an MC-max model of depth L. Then, for every database $\mathcal{D}$, $T_{\mathcal{R}_{\mathcal{M}}^{\max}}(\mathcal{D}) = T_{\mathcal{M}}(\mathcal{D})$ when $\gamma=\beta$. Moreover, the program $\mathcal{R}_{\mathcal{M}}^{\max}$ contains at most $\mathcal{O}\big(\delta^{L+1}\cdot\nu^{2L+2}\big)$ rules, where $\nu = \max_{1\le k\le\delta}|P_k|$.*

Unlike the MC bound in Theorem 1, this bound is independent of path cardinalities: extraction enumerates schemas rather than combinations of path counts.

6 Evaluation

We evaluate our models on the tabular datasets WP-IND, JF-IND, and MFB-IND (Yadati, 2020) under the inductive setting, where constants in the test sets may not appear during training. To assess performance on binary relations, we additionally use inductive datasets from Teru et al. (2020), derived from FB15k-237 (Toutanova & Chen, 2015), NELL-995 (Xiong et al., 2017), and WN18RR (Dettmers et al., 2018), following the original data splits. We compare against the hypergraph-based baselines HyperGCN (Yadati et al., 2019), GMPNN (Yadati, 2020), and HC-MPNN (Huang et al., 2025), as well as LLM-based approaches: Chain-of-Thought (CoT) prompting with GPT-5 mini (OpenAI, 2025) and TabLLM (Hegselmann et al., 2023). Further details on datasets and experimental settings are provided in Appendix F.

We also implement rule extraction for MC and MC-max models. Although presented declaratively, all extracted programs were computed explicitly by enumerating the corresponding candidate rules and threshold filtering by the definitions in Sections 4 and 5. In practice, the constructions are tractable due to the bounded cardinalities and the sparsity of realized path schemas in the datasets.

Table 1: Results (%) of Tabular Cell Completion (mean±standard deviation for non-LLM models).

	WP-IND					JF-IND					MFB-IND				
	P	R	Acc	PR-AUC	F1	P	R	Acc	PR-AUC	F1	P	R	Acc	PR-AUC	F1
HyperGCN	54.8±2.2	51.9±7.8	54.3±1.6	53.4±2.8	53.0±2.8	61.7±1.5	53.7±3.6	60.2±1.1	57.7±1.8	57.4±2.1	60.5±0.8	83.3±3.2	64.5±0.6	71.9±1.3	70.1±0.7
GMPNN-sum	50.7±1.0	51.3±1.8	50.7±1.0	41.3±0.7	51.0±1.2	54.9±3.2	72.2±7.6	56.6±4.4	56.4±5.7	62.3±4.9	67.2±2.2	79.5±3.2	70.3±1.7	67.9±2.4	72.8±1.4
GMPNN-mean	51.8±1.1	**93.7**±2.2	53.2±2.0	55.3±0.8	66.7±1.3	51.6±0.8	**91.4**±5.2	52.8±1.3	60.5±1.7	65.9±1.4	70.2±1.4	82.0±1.9	73.6±0.7	73.9±0.5	75.6±0.3
GMPNN-max	52.9±1.1	84.2±0.9	54.7±1.6	60.3±1.7	65.0±0.8	53.6±1.1	80.6±7.4	55.4±1.6	63.2±2.2	64.3±2.4	80.7±0.3	**92.2**±1.2	85.1±0.6	88.8±0.4	86.1±0.6
HC-MPNN	64.2±3.2	85.9±6.0	68.5±2.3	75.3±0.9	**73.2**±2.2	75.0±3.4	62.3±5.8	70.7±1.2	75.7±0.4	**68.0**±2.8	86.5±0.6	92.2±1.5	88.9±0.5	91.3±1.1	89.2±0.6
CoT (GPT-5 mini, 3-shot)	86.8	29.2	62.4	-	43.7	96.4	40.1	69.3	-	56.6	84.6	11.1	54.5	-	19.6
CoT (GPT-5 mini, 5-shot)	87.7	28.9	62.4	-	43.5	**96.7**	43.1	70.8	-	59.6	85.7	12.1	55.0	-	21.2
TabLLM (Llama 3.1)	86.9	31.7	63.5	-	46.5	79.9	54.1	70.2	-	64.5	80.0	6.4	52.4	-	11.9
MC ($L=2, N=1$)	80.1±1.8	51.1±0.7	69.2±0.6	72.8±0.2	62.4±0.5	78.2±2.6	55.8±0.8	70.0±0.9	**76.6**±0.3	65.1±0.5	91.6±1.1	79.1±2.8	85.9±0.9	90.5±1.1	84.9±1.3
MC ($L=2, N=2$)	80.3±2.2	51.4±0.8	69.4±0.6	72.9±0.3	62.7±0.3	74.9±2.0	56.0±1.4	68.5±0.6	76.3±0.2	64.1±0.4	89.9±3.5	82.1±5.6	86.3±1.2	90.2±0.3	85.6±1.7
MC ($L=2, N=3$)	80.4±1.3	51.1±0.6	69.3±0.5	73.0±0.3	62.5±0.4	75.7±2.3	56.2±1.2	69.0±0.8	76.2±0.3	64.5±0.5	91.3±1.1	81.5±3.5	86.8±1.3	91.1±1.3	86.1±1.7
MC-max ($L=2, N=1$)	**88.1**±3.9	49.0±2.6	70.3±0.5	73.4±0.5	62.2±1.3	82.5±2.5	53.1±3.8	70.8±1.1	71.9±2.6	64.5±2.4	91.2±1.1	85.0±3.6	88.4±1.3	88.7±1.7	87.9±1.6
MC-max ($L=2, N=2$)	85.3±3.3	49.7±2.0	70.5±0.8	73.2±0.3	62.8±1.3	81.7±2.7	55.8±2.6	71.5±1.0	74.6±1.6	66.2±1.5	**92.7**±0.7	84.4±3.8	88.9±1.5	91.6±0.7	88.3±1.9
MC-max ($L=2, N=3$)	83.9±2.1	50.5±1.2	70.4±0.7	73.4±0.6	63.0±0.9	83.6±2.1	54.4±2.5	**71.8**±0.6	74.9±2.1	65.8±1.3	92.3±0.4	87.8±1.5	**90.2**±0.5	**92.8**±0.8	**90.0**±0.6
MC-max ($L=3, N=1$)	80.3±4.5	57.8±3.0	71.7±1.7	76.4±1.6	67.1±1.8	79.3±4.7	55.7±5.3	70.3±1.0	73.1±1.5	65.2±2.7	86.8±2.4	66.8±9.2	78.3±3.9	85.1±1.7	75.2±5.7
MC-max ($L=3, N=2$)	82.4±4.6	58.2±2.4	72.7±1.4	**76.8**±0.8	68.1±0.8	81.5±3.0	54.8±2.7	71.0±0.8	72.9±3.0	65.4±1.3	89.9±2.9	79.3±3.3	85.1±1.6	89.0±2.2	84.2±1.7
MC-max ($L=3, N=3$)	82.9±1.3	58.9±0.8	**73.4**±0.4	76.5±0.9	68.8±0.4	77.3±2.1	60.5±1.4	71.3±1.3	73.5±1.9	67.9±1.3	90.0±1.2	88.0±2.9	89.1±0.8	90.8±1.2	88.9±1.0

Table 2: Rules Learned by MC-max from Tabular Datasets.

WP-IND	$\texttt{Deceased-Place-Country}(x_1,x,y) \leftarrow \texttt{Deceased-Place-Country}^3(x_1,x) \wedge \texttt{Deceased-Place-Country}(w_1,x,y)$
	$\texttt{Member-Membership-Role}(x_1,x,y) \leftarrow \texttt{Member-Membership-Role}^3(x_1,x) \wedge \texttt{Member-Membership-Role}(w_1,x,y)$
	$\texttt{Person-Sibling-Kinship}(x,y,x_2) \leftarrow \texttt{Person-Sibling-Kinship}^2(x,x_2) \wedge \texttt{Person-Sibling-Kinship}(y,x,w_1)$
	$\texttt{Politician-Position-Predecessor-Successor}(x,x_2,y,x_3) \leftarrow$
	$\quad \texttt{Politician-Position-Predecessor-Successor}^3(x,x_2,x_3) \wedge \texttt{Politician-Position-Predecessor-Successor}(y,w_1,w_2,x)$
	$\texttt{Politician-Position-Predecessor-Successor}(x,x_1,y,x_2) \leftarrow \texttt{Politician-Position-Predecessor-Successor}^3(x,x_1,x_2)$
	$\quad \wedge \texttt{Politician-Position-Predecessor-Successor}(x,w_1,w_2,z_1) \wedge \texttt{Politician-Position-Predecessor-Successor}(z_2,w_3,w_4,z_1)$
	$\quad \wedge \texttt{Politician-Position-Predecessor-Successor}(y,w_5,w_6,z_2)$
JF-IND	$\texttt{Country-Player-Olympics}(y,x,x_2) \leftarrow \texttt{Country-Player-Olympics}^1(x,x_2) \wedge \texttt{Country-Player-Olympics}(y,x,w_1)$
	$\texttt{Player-Event-Player}(x_1,y,x) \leftarrow \texttt{Player-Event-Player}^2(x_1,x) \wedge \texttt{Player-Event-Player}(x,y,w_1)$
	$\texttt{Player-Event-Team}(x,x_2,y) \leftarrow \texttt{Player-Event-Team}^3(x,x_2) \wedge \texttt{Team-Player-Event}(y,x,w_1)$
	$\texttt{Team-Player-Event}(x,x_2,y) \leftarrow \texttt{Team-Player-Event}^3(x,x_2) \wedge \texttt{Team-Player-Event}(x,w_1,y)$
MFB-IND	$\texttt{Ethnicity-Language-Person}(x_1,y,x) \leftarrow$
	$\quad \texttt{Ethnicity-Language-Person}^2(x_1,x) \wedge \texttt{Ethnicity-Language-Person}(z_1,w_1,x) \wedge \texttt{Ethnicity-Language-Person}(z_1,y,w_2)$
	$\texttt{Music-Artist-Place}(x_1,x,y) \leftarrow \texttt{Music-Artist-Place}^3(x_1,x) \wedge \texttt{Music-Artist-Place}(w_1,x,y)$
	$\texttt{Music-Genre-Subgenre}(y,x,x_1) \leftarrow \texttt{Music-Genre-Subgenre}^1(x,x_1) \wedge \texttt{Music-Genre-Subgenre}(w_1,x,z_1) \wedge \texttt{Music-Genre-Subgenre}(y,z_1,w_2)$
	$\texttt{Type-Singer-Instrument}(y,x,x_2) \leftarrow$
	$\quad \texttt{Type-Singer-Instrument}^1(x,x_2) \wedge \texttt{Type-Singer-Instrument}(w_1,x,z_1) \wedge \texttt{Type-Singer-Instrument}(y,w_2,z_1)$

6.1 Tabular Cell Completion

Table 1 reports Precision (P), Recall (R), Accuracy (Acc), Area Under the Precision–Recall Curve (PR-AUC), and F1 score. PR-AUC is omitted for LLM methods, as they do not produce continuous ranking scores for PR-area computation. Overall, MC and MC-max achieve strong performance, with our models attaining the best accuracy and PR-AUC on all the datasets. MC-max slightly exceeds MC in accuracy and F1 despite its limited expressivity, suggesting that selectively aggregating path evidence can be effective for cell completion. Among the baselines, HC-MPNN emerges as the strongest, though unable to provide faithful explanations for its predictions. GMPNN and HC-MPNN generally achieve higher recall, whereas our models offer better precision than hypergraph baselines. LLM baselines tend towards conservative predictions, often missing valid completions.

Comparison of MC and MC-max. Table 3 provides an internal comparison on binary relations. MC and MC-max demonstrated complementary strengths: MC achieves higher precision and accuracy, while MC-max yields higher recall. We attribute this to their aggregation mechanisms. The sum aggregation in MC considers all paths, including potentially irrelevant ones, leading to more fine-grained yet conservative predictions. In contrast, the max aggregation in MC-max focuses on the most salient path. These results suggest MC-max's practical usability despite its simpler design.

Training Time. MC and MC-max exhibit training times ranging from minutes to hours, indicating their practical feasibility. More details are provided in Appendix G.

6.2 Rule Extraction

Extracted Rules. Table 2 presents examples of top-weighted rules extracted by MC-max from the WP-IND, JF-IND, and MFB-IND datasets, where weights correspond to the learned path scores used by the model. Most rules demonstrate binary relationships, such as the first two rules from WP-IND. In particular, the model was able to capture symmetrical relationships, exemplified by the third rule from WP-IND (i.e., between `Person` and `Sibling`), as well as transitive relationships, such

Table 3: Results (%) of MC (M) and MC-max (m) on Binary Datasets.

		P		R		Acc		PR-AUC		F1	
		M	m	M	m	M	m	M	m	M	m
FB15k-237	V1	**53.0**	42.2	51.0	**65.4**	**52.5**	37.8	**66.8**	66.3	**51.8**	51.3
	V2	**67.7**	56.5	61.8	**72.7**	**65.5**	58.0	77.1	**78.5**	**64.3**	63.4
	V3	**76.7**	68.4	58.1	**62.3**	**70.2**	66.6	**78.6**	77.7	**66.1**	65.1
	V4	**78.7**	66.4	56.4	**68.1**	**70.4**	66.7	79.0	**79.6**	65.6	**67.2**
NELL-995	V1	**92.1**	72.0	**98.8**	95.9	**95.1**	77.7	**98.8**	94.6	**95.3**	81.7
	V2	74.1	**77.6**	**69.1**	68.6	72.0	**74.1**	82.7	**84.0**	71.0	**72.5**
	V3	**74.4**	60.0	74.4	**84.5**	**74.3**	63.6	84.6	**87.2**	**74.3**	70.0
	V4	56.4	**59.8**	78.6	**78.9**	58.8	**62.1**	80.8	**82.2**	65.7	**67.7**
WN18RR	V1	**90.3**	86.0	65.7	**66.6**	**79.3**	77.6	**83.7**	82.3	**76.1**	75.0
	V2	**96.8**	86.5	62.8	**63.7**	**80.4**	76.7	**82.6**	81.1	**76.2**	73.3
	V3	71.6	**85.6**	**35.8**	29.0	60.8	**62.0**	**66.6**	62.7	**47.7**	43.3
	V4	**94.9**	92.8	**60.1**	59.6	**78.4**	77.5	**80.6**	80.2	**73.6**	72.6

Table 4: Rule Extraction Time (seconds) for MC-max.

		$N=1$	$N=2$	$N=3$
WP-IND		1.375	2.251	2.371
JF-IND		0.777	1.205	1.323
MFB-IND		0.025	0.030	0.039
FB15k-237	V1	54.87	100.9	109.6
	V2	71.75	138.0	159.8
	V3	39.30	84.83	135.8
	V4	78.28	95.78	142.1
NELL-995	V1	0.006	0.009	0.009
	V2	4.988	5.799	10.74
	V3	21.90	40.82	66.59
	V4	3.634	6.998	10.51
WN18RR	V1	0.001	0.001	0.002
	V2	0.002	0.002	0.003
	V3	0.001	0.002	0.003
	V4	0.001	0.001	0.002

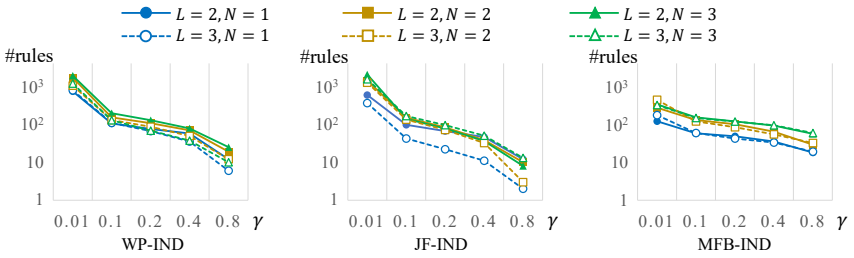


Figure 3: Number of Rules Extracted under Different γ.

as the third rule from MFB-IND (i.e., `Genre` and `Subgenre`). Additionally, it demonstrates the ability to capture schema alignment across related tables, such as the third rule from JF-IND, where it aligns the positions of `Team` and `Player` across different predicates `Player-Event-Team` and `Team-Player-Event`. Increasing the depth L enables the model to capture longer and more complex dependencies, but may also introduce semantically redundant rules. In practice, L can be adjusted based on the observed path-length distribution to balance expressivity and rule conciseness.

Number of Rules. Figure 3 shows the size of extracted rule sets under different thresholds. Overall, the number of rules ranges from tens to a few thousand, with an average of at most 10–20 rules per head atom. This suggests that the resulting rule sets remain of manageable size for inspection. By adjusting N, L, and γ, users can trade off between rule compactness and explanatory richness.

Rule Extraction Time. Table 4 reports the runtime of rule extraction for MC-max models with $L = 3$. Generally, runtime increased with rank N but remained below 3 minutes in all cases. For dataset-specific rule extraction for MC models, the average runtime remained below 5 minutes per 100 predicted facts on each test set. These results indicate the practical feasibility of our approaches.

7 Conclusion and Future Work

In this work, we introduced interpretable models for tabular cell completion that combine strong empirical performance with faithful rule extraction. Our approach provides human-readable rules that exactly replicate the outputs of the model, offering a precise and verifiable form of explanation. While our work focuses on path-based relational patterns, an important direction for future work is to extend rule learning to more complex structures, such as multi-branch or cyclic rules, which could capture richer dependencies among multiple attributes. Another promising direction concerns evaluation methodology. Existing benchmarks for tabular data completion largely rely on random data splits, which may not adequately test a model's ability to learn and generalize complex rules. We will develop more informative benchmarks based on rule-driven fact generation (Liu et al., 2023), enabling controlled evaluation of rule learning performance under increasingly complex patterns.

AI USE STATEMENT

Generative AI tools were used for minor language polishing of the manuscript. The authors take full responsibility for all content, including the writing, analysis, and results presented in the paper.

ETHICS STATEMENT

We have reviewed the ethics guidelines and we believe our research complies with the ICLR Code of Ethics at `https://iclr.cc/public/CodeOfEthics` in every respect. No human subjects or sensitive data were involved, and we do not foresee any ethical, privacy, or fairness concerns arising from this research.

REPRODUCIBILITY STATEMENT

The proofs of all the lemmas and theorems are provided in Appendix B. The datasets and source codes used in our experiments are available at the anonymous GitHub repository with documentation: `https://anonymous.4open.science/r/HARL`.

REFERENCES

Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995. ISBN 0-201-53771-0.

Ahmad Ahmadov, Maik Thiele, Julian Eberius, Wolfgang Lehner, and Robert Wrembel. Towards a hybrid imputation approach using web tables. In *Proc. 2nd IEEE/ACM Int. Symp. on Big Data Computing (BDC 2015)*, pp. 21–30, 2015. doi: 10.1109/BDC.2015.38.

Can Chen and Yangyu Liu. A survey on hyperlink prediction. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. doi: 10.1109/TNNLS.2023.3286280.

Tianlang Chen, Charilaos I. Kanatsoulis, and Jure Leskovec. RelGNN: Composite message passing for relational deep learning. In *42nd Int. Conf. on Machine Learning (ICML 2025)*, 2025.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. Binding language models in symbolic languages. In *11th Int. Conf. on Learning Representations, (ICLR 2023)*, 2023.

Fei Chiang and Renée J. Miller. Discovering data quality rules. *Proc. VLDB Endow.*, 1(1):1166–1177, 2008. doi: 10.14778/1453856.1453980.

Xu Chu, Ihab F. Ilyas, Paolo Papotti, and Yin Ye. RuleMiner: Data quality rules discovery. In *Proc. of the 30th Int. Conf. on Data Engineering (ICDE 2014)*, pp. 1222–1225, 2014. doi: 10.1109/ICDE.2014.6816746.

William W. Cohen, Fan Yang, and Kathryn Mazaitis. Tensorlog: A probabilistic database implemented using deep-learning infrastructure. *Journal of Artificial Intelligence Research*, 67:285–325, 2020. doi: 10.1613/JAIR.1.11944.

Evgeny Dantsin, Thomas Eiter, Georg Gottlob, and Andrei Voronkov. Complexity and expressive power of logic programming. *ACM Computing Surveys*, 33(3):374–425, 2001. doi: 10.1145/502807.502810.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2D knowledge graph embeddings. In *Proc. of the 32nd AAAI Conf. on Artificial Intelligence (AAAI 2018)*, pp. 1811–1818, 2018. doi: 10.1609/AAAI.V32I1.11573.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In *Proc. of the Annual Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023.

Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research*, 61:1–64, 2018. doi: 10.1613/JAIR.5714.

Bahare Fatemi, Perouz Taslakian, David Vázquez, and David Poole. Knowledge hypergraphs: Prediction beyond binary relations. In *Proc. of the 29th Int. Joint Conf. on Artificial Intelligence (IJCAI 2020)*, pp. 2191–2197, 2020. doi: 10.24963/IJCAI.2020/303.

João Ferreira, Manuel de Sousa Ribeiro, Ricardo Gonçalves, and João Leite. Looking inside the black-box: Logic-based explanations for neural networks. In *Proc. of the 19th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2022)*, 2022.

Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Position: Relational deep learning - graph representation learning on relational databases. In *41st Int. Conf. on Machine Learning (ICML 2024)*. OpenReview.net, 2024.

Sumit Gulwani. Automating string processing in spreadsheets using input-output examples. In *Proc. of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2011)*, pp. 317–330, 2011. doi: 10.1145/1926385.1926423.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David A. Sontag. TabLLM: Few-shot classification of tabular data with large language models. In *Int. Conf. on Artificial Intelligence and Statistics (AISTATS 2023)*, volume 206 of *Proceedings of Machine Learning Research*, pp. 5549–5581, 2023.

Xingyue Huang, Miguel A. Romero Orth, Pablo Barceló, Michael M. Bronstein, and .Ismail .Ilkan Ceylan. Link prediction with relational hypergraphs. *Transactions on Machine Learning Research*, 2025, 2025.

Samuel Kolb, Sergey Paramonov, Tias Guns, and Luc De Raedt. Learning constraints in spreadsheets and tabular data. *Machine Learning*, 106(9-10):1441–1468, 2017. doi: 10.1007/S10994-017-5640-X.

Shuwen Liu, Bernardo Cuenca Grau, Ian Horrocks, and Egor V. Kostylev. Revisiting inferential benchmarks for knowledge graph completion. In *Proc. of the 20th International Conference on Principles of Knowledge Representation and Reasoning (KR 2023)*, pp. 461–471, 2023. doi: 10.24963/KR.2023/45.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 4765–4774, 2017.

Meta. Llama 3.1. Model released under the Llama 3.1 Community License, 2024. Pretrained large language model (8B / 70B / 405B parameters).

OpenAI. GPT-5 mini. `https://openai.com`, 2025. Large language model.

Meng Qu, Junkun Chen, Louis-Pascal A. C. Xhonneux, Yoshua Bengio, and Jian Tang. RNNLogic: Learning logic rules for reasoning on knowledge graphs. In *Proc. of the 9th Int. Conf. on Learning Representations (ICLR 2021)*, 2021.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. DRUM: end-to-end differentiable rule mining on knowledge graphs. In *Proc. of the Annual Conf. on Neural Information Processing Systems (NeurIPS 2019)*, pp. 15321–15331, 2019.

Daniel J Stekhoven and Peter Bühlmann. MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Yige Sun, Jing Li, Yifan Xu, Tingting Zhang, and Xiaofeng Wang. Deep learning versus conventional methods for missing data imputation: A review and comparative study. *Expert Systems with Applications*, 227:120201, 2023. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2023.120201.

David Jaime Tena Cucala, Bernardo Cuenca Grau, Egor V. Kostylev, and Boris Motik. Explainable GNN-based models over knowledge graphs. In *Proc. of the 10th Int. Conf. on Learning Representations (ICLR 2022)*, 2022a.

David Jaime Tena Cucala, Bernardo Cuenca Grau, and Boris Motik. Faithful approaches to rule learning. In *Proc. of the 19th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2022)*, 2022b.

David Jaime Tena Cucala, Bernardo Cuenca Grau, Boris Motik, and Egor V. Kostylev. On the correspondence between monotonic max-sum gnns and datalog. In *Proc. of the 20th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2023)*, pp. 658–667, 2023. doi: 10.24963/KR.2023/64.

Komal K. Teru, Etienne G. Denis, and William L. Hamilton. Inductive relation prediction by subgraph reasoning. In *Proc. of the 37th Int. Conf. on Machine Learning (ICML 2020)*, pp. 9448–9457, 2020.

Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proc. of the 3rd Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2015)*, pp. 57–66, 2015. doi: 10.18653/V1/W15-4007.

Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.

Chenxu Wang, Xin Wang, Zhao Li, Zirui Chen, and Jianxin Li. Hyconve: A novel embedding model for knowledge hypergraph link prediction with convolutional neural networks. In *Proc. of the ACM Web Conf. (WWW 2023)*, pp. 188–198, 2023a. doi: 10.1145/3543507.3583256.

Xiaxia Wang, David Jaime Tena Cucala, Bernardo Cuenca Grau, and Ian Horrocks. Faithful rule extraction for differentiable rule learning models. In *Proc. of the 12th Int. Conf. on Learning Representations (ICLR 2024)*, 2024a.

Xinyu Wang, Isil Dillig, and Rishabh Singh. Synthesis of data completion scripts using finite tree automata. *Proc. ACM Program. Lang.*, 1(OOPSLA):62:1–62:26, 2017. doi: 10.1145/3133886.

Zihao Wang, Yangqiu Song, Ginny Y. Wong, and Simon See. Logical message passing networks with one-hop inference on atomic formulas. In *Proc. of the 11th Int. Conf. on Learning Representations (ICLR 2023)*, 2023b.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and Tomas Pfister. Chain-of-table: Evolving tables in the reasoning chain for table understanding. In *12th Int. Conf. on Learning Representations, (ICLR 2024)*, 2024b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proc. of the Annual Conf. on Neural Information Processing Systems (NeurIPS 2022)*, 2022.

Wenhan Xiong, Thien Hoang, and William Yang Wang. DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proc. of the 2017 Conf. on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pp. 564–573, 2017. doi: 10.18653/V1/D17-1060.

Naganand Yadati. Neural message passing for multi-relational ordered and recursive hypergraphs. In *Proc. of the Annual Conf. on Neural Information Processing Systems (NeurIPS 2020)*, 2020.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha P. Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. In *Proc. of the Annual Conf. on Neural Information Processing Systems (NeurIPS 2019)*, pp. 1509–1520, 2019.

Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proc. of the ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 2012)*, pp. 97–108, 2012. doi: 10.1145/2213836.2213848.

Fan Yang, Zhilin Yang, and William W. Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *Proc. of the Annual Conf. on Neural Information Processing Systems (NeurIPS 2017)*, pp. 2319–2328, 2017.

Shichang Zhang, Jiani Zhang, Xiang Song, Soji Adeshina, Da Zheng, Christos Faloutsos, and Yizhou Sun. PaGE-Link: Path-based graph neural network explanation for heterogeneous link prediction. In *Proc. of the ACM Web Conf. 2023 (WWW 2023)*, pp. 3784–3793, 2023. doi: 10.1145/3543507.3583511.

Shuo Zhang and Krisztian Balog. Auto-completion for data cells in relational tables. In *Proc. of the 28th ACM Int. Conf. on Information and Knowledge Management (CIKM 2019)*, pp. 761–770, 2019. doi: 10.1145/3357384.3357932.

A Extended Related Work

Rule Learning. Rule learning aims to automatically provide rules that explain the predictions of ML models. A number of approaches (Yang et al., 2017; Evans & Grefenstette, 2018; Sadeghian et al., 2019; Qu et al., 2021; Ferreira et al., 2022; Zhang et al., 2023; Wang et al., 2023b) have been proposed to extract such rules, but many of them lack formal guarantees establishing the relationship between the model and the extracted rules. Instead, they often rely on informal claims that the rules "approximate" the model's behavior (Yang et al., 2017; Evans & Grefenstette, 2018; Sadeghian et al., 2019; Qu et al., 2021).

To fill the gap, *faithful rule learning* (Tena Cucala et al., 2022a;b; 2023; Wang et al., 2024a) has been investigated to not only provide rule-based explanation, but also ensure theoretical equivalence between the model and the extracted rules. However, these methods are typically restricted to binary relational data (Tena Cucala et al., 2022b; Wang et al., 2024a), such as knowledge graphs. Extending them to tabular data is nontrivial due to the more complex relations among data cells described by each row. To the best of our knowledge, no prior work has investigated faithful rule learning for tabular cell completion.

Data Quality Rule Discovery. *Data quality rule discovery* focuses on extracting statistical dependencies that profile tables or detect inconsistencies (Chiang & Miller, 2008; Chu et al., 2014). These methods typically take a table as input and output rules describing frequent patterns, optimizing for metrics such as rule quality or confidence. Our work differs from data quality rule discovery in multiple aspects. Despite the different input, the extracted rules in our work serve as an explanatory mechanism for ML models, rather than as stand-alone quality indicators, and directly support the task of completing missing cells in multi-relational databases.

Learning over Higher-Arity Relations. Our task is closely related to *hypergraph link prediction*, sometimes also referred to as *n-ary knowledge base completion* (Chen & Liu, 2023; Fatemi et al., 2020; Yadati et al., 2019; Wang et al., 2023a; Huang et al., 2025), since a database can be viewed as a hypergraph. Recent ML approaches extend *Graph Neural Networks (GNNs)* and propose *hypergraph neural networks* (Yadati et al., 2019; Yadati, 2020), where tabular data is represented as a hypergraph with cell values as nodes and rows as hyperedges. While effective, these methods generally lack interpretability, motivating our focus on explainable alternatives. Another line of work is *relational deep learning* (Fey et al., 2024; Chen et al., 2025). These approaches adapt GNNs to relational databases but they represent a database differently: rows are nodes and edges are defined by primary–foreign key relations. In contrast, both hypergraph neural networks (Yadati et al., 2019; Yadati, 2020) and our approach operate at the *cell* level, treating individual data values as nodes, which is more suitable for cell completion tasks.

LLM-based Table Understanding. Recent works have explored the use of *large language models* for table understanding and reasoning (Hegselmann et al., 2023; Cheng et al., 2023; Wang et al., 2024b). However, as generative models may produce hallucinated values and cannot provide faithful explanations for their results, such uncertainty is especially problematic in safety-critical or legally regulated settings. In contrast, our approach offers provably faithful explanations for the results.

B Proofs

B.1 Proof of Lemma 1

Lemma 1. *Let $\mathcal{M}$ be an MC model with rank N and depth L as in Definition 2. Let $P_h(\mathbf{c})$ be an incomplete fact in $\mathcal{D}$ with a null value in position $1 \leq t \leq |P_h|$. Then, $\mathbf{v}_{\mathbf{c}}^{h,t}$ can be written as*

$$\mathbf{v}_{\mathbf{c}}^{h,t} = \sum_{\omega \in \Omega} \mathtt{wt}_{\mathcal{M}}^{h,t}(\omega) \cdot \mathbf{v}_{\omega,\mathcal{D},\mathbf{c}} \,,$$

where $\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}$ is a non-negative vector dependent only on the data (and independent of the model parameters) defined as follows: (1) if $\omega = \top$, then $\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}(u)$ counts the occurrences of c_u in $\mathbf{c}$; (2) if ω is a path schema of length $\ell \geq 1$, then $\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}(u)$ counts the distinct paths in $\mathcal{D}$ matching ω

and connecting constants in $\mathbf{c}$ *to* c_u*. Moreover, the weight* $\mathtt{wt}^{h,t}_{\mathcal{M}}(\omega)$ *is given by*

$$\mathtt{wt}^{h,t}_{\mathcal{M}}(\omega) = \sum_{(d_1,\ldots,d_L)\in S^L_\omega} \sum_{i=1}^{N} \prod_{j=1}^{L} \mathbf{b}^{h,t}(i,j,d_j)\,, \tag{2}$$

where $S^L_\omega \subseteq \{1,\ldots,K\}^L$ *is the set of all sequences* $(d_1,\ldots,d_L)$ *obtained by placing the steps of* ω *in order into* L *positions and filling the remaining positions with the symbol* K *(corresponding to* $\top$*). In particular, for* $\omega = \top$*, we have* $S^L_\top = \{(K,\ldots,K)\}$*.*

Proof. Let $\mathcal{D}$ be a database over constants $c_1,\ldots,c_\epsilon$. For simplicity, write $\hat{\mathbf{M}}_d = \mathbf{M}^k_{p,q}$ when $d = \mathtt{d}_{k,p,q}$ for $1 \le d \le K-1$, and let $\hat{\mathbf{M}}_K$ be the $\epsilon\times\epsilon$ identity matrix. Then Equation (1) can be written as Equation (8).

$$(\mathbf{v}^{i,j}_{c_s})^\intercal = (\mathbf{v}^{i,j-1}_{c_s})^\intercal \cdot \sum_{d=1}^{K} \mathbf{b}^{h,t}(i,j,d)\cdot \hat{\mathbf{M}}_d\,. \tag{8}$$

Also, the computation of the result vector $\mathbf{v}^{h,t}_{\mathbf{c}}$ can be written as Equation (9).

$$(\mathbf{v}^{h,t}_{\mathbf{c}})^\intercal = \sum_{i=1}^{N}\sum_{c_s\in\mathbf{c}} (\mathbf{v}^{i,0}_{c_s})^\intercal \cdot \prod_{j=1}^{L}\left(\sum_{d=1}^{K} \mathbf{b}^{h,t}(i,j,d)\cdot\hat{\mathbf{M}}_d\right). \tag{9}$$

The distributive and associative properties of the sum and product operators in Equation (9) allow us to rewrite it as Equation (10).

$$(\mathbf{v}^{h,t}_{\mathbf{c}})^\intercal = \sum_{(d_1,\ldots,d_L)\in\{1,\ldots,K\}^L} \left(\sum_{i=1}^{N}\prod_{j=1}^{L}\mathbf{b}^{h,t}(i,j,d_j)\right)\cdot\left(\sum_{c_s\in\mathbf{c}}(\mathbf{v}^{1,0}_{c_s})^\intercal\right)\cdot\hat{\mathbf{M}}_{d_1}\cdot\hat{\mathbf{M}}_{d_2}\cdot\ldots\cdot\hat{\mathbf{M}}_{d_L}\,. \tag{10}$$

Each sequence $(d_1,\ldots,d_L)\in\{1,\ldots,K\}^L$ corresponds to a path schema or $\top$, as introduced in Definition 1. Let $(d'_1,\ldots,d'_\ell)$ be the subsequence obtained by removing all occurrences of K. If $\ell = 0$, the sequence corresponds to $\top$; if $\ell \ge 1$, each d'_i determines a unique triple (k_i,p_i,q_i) via the indexing $\mathtt{d}_{k_i,p_i,q_i}$, yielding the path schema $(P_{k_1},p_1,q_1),\ldots,(P_{k_\ell},p_\ell,q_\ell)$.

Using the set S^L_ω from Equation (2), Equation (10) can be rewritten as Equation (11).

$$(\mathbf{v}^{h,t}_{\mathbf{c}})^\intercal = \sum_{\omega\in\Omega}\left(\sum_{(d_1,\ldots,d_L)\in S^L_\omega}\left(\sum_{i=1}^{N}\prod_{j=1}^{L}\mathbf{b}^{h,t}(i,j,d_j)\right)\cdot\left(\sum_{c_s\in\mathbf{c}}(\mathbf{v}^{1,0}_{c_s})^\intercal\right)\cdot\hat{\mathbf{M}}_{d_1}\cdot\hat{\mathbf{M}}_{d_2}\cdot\ldots\cdot\hat{\mathbf{M}}_{d_L}\right). \tag{11}$$

Since $\hat{\mathbf{M}}_K$ is the identity, each vector $(\mathbf{v}^{i,0}_{c_s})^\intercal\cdot\hat{\mathbf{M}}_{d_1}\cdots\hat{\mathbf{M}}_{d_L}$ for $c_s\in\mathbf{c}$ equals $(\mathbf{v}^{i,0}_{c_s})^\intercal\cdot\hat{\mathbf{M}}_{d'_1}\cdots\hat{\mathbf{M}}_{d'_\ell}$. If $\ell \ge 1$, a simple induction shows that the u-th entry counts the paths of length ℓ from c_s to c_u matching the corresponding path schema (Definition 1). Summing over $c_s\in\mathbf{c}$, the u-th entry of $\left(\sum_{c_s\in\mathbf{c}}(\mathbf{v}^{i,0}_{c_s})^\intercal\right)\cdot\hat{\mathbf{M}}_{d_1}\cdots\hat{\mathbf{M}}_{d_L}$ gives the total count of such paths from constants in $\mathbf{c}$ to c_u. If $\ell = 0$, this vector reduces to $\sum_{c_s\in\mathbf{c}}(\mathbf{v}^{i,0}_{c_s})^\intercal$, whose u-th entry counts the occurrences of c_u in $\mathbf{c}$. In both cases, the vector equals $\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}$ as defined in the lemma statement, and is the same for all $(d_1,\ldots,d_L)\in S^L_\omega$ of a given ω. Replacing it in Equation (11) yields $(\mathbf{v}^{h,t}_{\mathbf{c}})^\intercal = \sum_{\omega\in\Omega}\mathtt{wt}^{h,t}_{\mathcal{M}}(\omega)\cdot\mathbf{v}_{\omega,\mathcal{D},\mathbf{c}}$, with

$$\mathtt{wt}^{h,t}_{\mathcal{M}}(\omega) = \sum_{(d_1,\ldots,d_L)\in S^L_\omega}\left(\sum_{i=1}^{N}\prod_{j=1}^{L}\mathbf{b}^{h,t}(i,j,d_j)\right). \tag{12}$$

□

B.2 Proof of Theorem 1

Theorem 1. *The program* $\mathcal{R}_{\mathcal{M}}$ *is faithful to* $\mathcal{M}$ *when* $\gamma = \beta$*, with* $|\mathcal{R}_{\mathcal{M}}| \le \delta\nu\big[2(C+1)^{|\Omega|-1}\big]^{\nu-1}$*, where* $\nu = \max_{1\le k\le\delta}|P_k|$ *and* $C = \max_{h,t}\max_{\omega\in\Omega,\,\omega\neq\top}\mathring{C}^{h,t}(\omega)$*.*

Proof. **Soundness.** We first prove soundness, namely that $T_{\mathcal{R}_\mathcal{M}}(\mathcal{D}) \subseteq T_\mathcal{M}(\mathcal{D})$ for each database $\mathcal{D}$. Let

$$P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_{\mathcal{R}_\mathcal{M}}(\mathcal{D}).$$

Then some rule $\varrho \in \mathcal{R}_\mathcal{M}$ and some grounding σ satisfy the body of ϱ in $\mathcal{D}$, with head instantiated to the fact above. For each $r \neq t$, the corresponding body conjunction φ_r is either $\top$ or a conjunction of multichain patterns that is satisfied by σ.

For each $r \neq t$, let $e_r = 1$ if the rule identifies x'_r with y, and $e_r = 0$ otherwise. Since the rule body is satisfied, $e_r = 1$ implies $c_{s_r} = c_u$. Moreover, for every $\omega \in \Omega \setminus \{\top\}$,

$$\texttt{count}_\mathcal{D}(\omega, c_{s_r}, c_u) \geq C_\omega(\varphi_r)\,,$$

where omitted patterns have cardinality zero. Here C_ω is unchanged by identifying the endpoints. By Definition 6,

$$\texttt{score}^{h,t}_\mathcal{M}(\varphi_r) = \sum_{\omega \in \Omega \setminus \{\top\}} C_\omega(\varphi_r)\, \texttt{wt}^{h,t}_\mathcal{M}(\omega) + e_r\, \texttt{wt}^{h,t}_\mathcal{M}(\top)\,.$$

By Lemma 1, the contribution of position r to the score for c_u is

$$\sum_{\omega \in \Omega \setminus \{\top\}} \texttt{wt}^{h,t}_\mathcal{M}(\omega)\, \texttt{count}_\mathcal{D}(\omega, c_{s_r}, c_u) + \mathbb{1}[c_{s_r} = c_u]\, \texttt{wt}^{h,t}_\mathcal{M}(\top)\,.$$

Since all weights are nonnegative and $e_r \leq \mathbb{1}[c_{s_r} = c_u]$, this contribution is at least $\texttt{score}^{h,t}_\mathcal{M}(\varphi_r)$.

Summing over all $r \neq t$, we obtain

$$\mathbf{v}^{h,t}_\mathbf{c}(u) \geq \texttt{score}^{h,t}_\mathcal{M}(\varrho).$$

Finally, since $\varrho \in \mathcal{R}_\mathcal{M}$, Definition 6 implies $\texttt{score}^{h,t}_\mathcal{M}(\varrho) > \gamma$. With $\gamma = \beta$, we conclude $\mathbf{v}^{h,t}_\mathbf{c}(u) > \beta$, so the model outputs the same completed fact. Therefore, $P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_\mathcal{M}(\mathcal{D})$.

Completeness. We show that $T_\mathcal{M}(\mathcal{D}) \subseteq T_{\mathcal{R}_\mathcal{M}}(\mathcal{D})$ for each database $\mathcal{D}$. Let $P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_\mathcal{M}(\mathcal{D})$. Then, $\mathbf{v}^{h,t}_\mathbf{c}(u) > \beta$.

For each $r \neq t$ and $\omega \in \Omega \setminus \{\top\}$, we choose the cardinality

$$C_\omega(\varphi_r) = \min\big(\texttt{count}_\mathcal{D}(\omega, c_{s_r}, c_u), \mathring{C}^{h,t}(\omega)\big)\,,$$

omitting patterns of cardinality zero. Let $x'_r = y$ if $c_{s_r} = c_u$, and $x'_r = x_r$ otherwise. These choices define conjunctions $\varphi_r \in \Theta^{h,t}$ with scores

$$\texttt{score}^{h,t}_\mathcal{M}(\varphi_r) = \sum_{\omega \in \Omega \setminus \{\top\}} \texttt{wt}^{h,t}_\mathcal{M}(\omega)\, \min\big(\texttt{count}_\mathcal{D}(\omega, c_{s_r}, c_u), \mathring{C}^{h,t}(\omega)\big) + \mathbb{1}[c_{s_r} = c_u]\, \texttt{wt}^{h,t}_\mathcal{M}(\top)\,.$$

Let ϱ be the corresponding rule of the form (4). Since the model score exceeds $\beta \geq 0$, some positive contribution exists: either a nonempty path pattern is included or a source variable is identified with y. Thus ϱ satisfies Definition 5.

If a positive-weight schema has a path count exceeding its cardinality bound at some source position r, then Equation (3) gives $\texttt{score}^{h,t}_\mathcal{M}(\varphi_r) > \beta$. Otherwise, truncation leaves all positive-weight contributions unchanged, so Lemma 1 gives

$$\texttt{score}^{h,t}_\mathcal{M}(\varrho) = \mathbf{v}^{h,t}_\mathbf{c}(u)\,.$$

In either case, $\texttt{score}^{h,t}_\mathcal{M}(\varrho) > \beta = \gamma$, so Definition 6 implies $\varrho \in \mathcal{R}_\mathcal{M}$.

Finally, the body of ϱ is satisfied in $\mathcal{D}$ by construction of the multichain patterns from the realized path counts. Therefore,

$$P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_{\mathcal{R}_\mathcal{M}}(\mathcal{D}).$$

Bound on rule number. Let $\nu = \max_{1 \le i \le \delta} |P_i|$. The number of path schemas in Ω is bounded by $|\Omega| = \mathcal{O}\big(\delta^L \nu^{2L}\big)$.

For each predicate P_h and position t, every conjunction in $\Theta^{h,t}$ is obtained by choosing, for each path schema $\omega \in \Omega \setminus \{\top\}$, a cardinality in $\{0, \dots, \mathring{C}^{h,t}(\omega)\}$ together with one of the two variants φ or $\varphi\{x \mapsto y\}$. Hence, $|\Theta^{h,t}| \le 2(C+1)^{|\Omega|-1}$, where $C = \max_{h,t} \max_{\omega \neq \top} \mathring{C}^{h,t}(\omega)$.

An MC rule selects one conjunction from $\Theta^{h,t}$ for each of the $|P_h| - 1 \le \nu - 1$ non-target positions. Therefore,

$$|\mathcal{R}_{\mathcal{M}}^{h,t}| \le \left[2(C+1)^{(|\Omega|-1)}\right]^{(\nu-1)}.$$

Finally, the number of (h,t) pairs is $\sum_{k=1}^{\delta} |P_k| \le \delta\nu$, so $|\mathcal{R}_{\mathcal{M}}| \le \delta\nu \cdot \left[2(C+1)^{(|\Omega|-1)}\right]^{(\nu-1)}$ which yields the stated bound.

□

B.3 Database-specific Rule Extraction

Here, we provide the formal construction of the database-specific program introduced in Section 4.3 and prove its formal guarantees.

Definition 10. *For a database $\mathcal{D}$, a path schema $\omega \in \Omega \setminus \{\top\}$, and constants c_s, c_u in $\mathcal{D}$, the* path count $\texttt{count}_{\mathcal{D}}(\omega, c_s, c_u)$ *is the number of distinct paths in $\mathcal{D}$ matching ω from c_s to c_u. We say that ω is* realized *between c_s and c_u if this count is positive.*

Definition 11. *For an MC model $\mathcal{M}$ and database $\mathcal{D}$, we construct a rule of the form* (4) *for each incomplete fact $P_h(c_{s_1}, \dots, c_{s_{t-1}}, ?, c_{s_{t+1}}, \dots, c_{s_{|P_h|}})$ in $\mathcal{D}$ and each replacement c_u predicted by $\mathcal{M}$. For each source position $r \neq t$ and path schema $\omega \in \Omega \setminus \{\top\}$, the conjunction $\varphi_r(x'_r, y)$ contains a multichain pattern with the corresponding core and cardinality*

$$C_\omega(\varphi_r) = \min\left(\texttt{count}_{\mathcal{D}}(\omega, c_{s_r}, c_u), \mathring{C}^{h,t}(\omega)\right),$$

where zero-cardinality patterns are omitted. Set $x'_r = y$ if $c_{s_r} = c_u$, and $x'_r = x_r$ otherwise. The database-specific program $\mathcal{R}_{\mathcal{M},\mathcal{D}}$ *consists of all rules obtained by this construction.*

Unlike the full program $\mathcal{R}_{\mathcal{M}}$ in Definition 6, this construction uses the observed path counts rather than enumerating all combinations of cardinalities.

Theorem 2. *When $\gamma = \beta$, the program $\mathcal{R}_{\mathcal{M},\mathcal{D}}$ satisfies $\mathcal{R}_{\mathcal{M},\mathcal{D}} \subseteq \mathcal{R}_{\mathcal{M}}$ and $T_{\mathcal{R}_{\mathcal{M},\mathcal{D}}}(\mathcal{D}) = T_{\mathcal{M}}(\mathcal{D})$. Moreover, the program $\mathcal{R}_{\mathcal{M},\mathcal{D}}$ can be constructed in time $\mathcal{O}\big(\epsilon^{L+\nu}\big)$, where $\nu = \max_{1 \le k \le \delta} |P_k|$, and ϵ is the number of constants in $\mathcal{D}$.*

Proof. We first prove that $\mathcal{R}_{\mathcal{M},\mathcal{D}} \subseteq \mathcal{R}_{\mathcal{M}}$. Let $\varrho \in \mathcal{R}_{\mathcal{M},\mathcal{D}}$ be a rule constructed according to Definition 11 for some fact

$$P_h(c_{s_1}, \dots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \dots, c_{s_{|P_h|}}) \in T_{\mathcal{M}}(\mathcal{D}).$$

For each position $r \neq t$, the conjunction $\varphi_r(x'_r, y)$ contains one multichain pattern for every path schema ω realized between c_{s_r} and c_u, with cardinality $\min\big(\texttt{count}_{\mathcal{D}}(\omega, c_{s_r}, c_u), \mathring{C}^{h,t}(\omega)\big)$. Hence every cardinality used in φ_r is bounded by $\mathring{C}^{h,t}(\omega)$. Assigning cardinality 0 to all non-realized path schemas therefore yields $\varphi_r \in \Theta^{h,t}$ by Definition 6.

The construction uses exactly the truncated cardinalities and endpoint identifications from the completeness proof of Theorem 1. If any positive-weight schema has an observed count exceeding its cardinality bound, its truncated contribution already exceeds β. Otherwise, truncation leaves all positive-weight contributions unchanged, including the zero-step contribution determined by the endpoint identifications. Thus, in either case, $\texttt{score}_{\mathcal{M}}^{h,t}(\varrho) > \beta = \gamma$. By Definition 6, $\varrho \in \mathcal{R}_{\mathcal{M}}$, and hence $\mathcal{R}_{\mathcal{M},\mathcal{D}} \subseteq \mathcal{R}_{\mathcal{M}}$. Consequently, $T_{\mathcal{R}_{\mathcal{M},\mathcal{D}}}(\mathcal{D}) \subseteq T_{\mathcal{R}_{\mathcal{M}}}(\mathcal{D}) \subseteq T_{\mathcal{M}}(\mathcal{D})$, where the second inclusion follows from the soundness part of Theorem 1.

To complete the proof, it remains to show

$$T_{\mathcal{M}}(\mathcal{D}) \subseteq T_{\mathcal{R}_{\mathcal{M},\mathcal{D}}}(\mathcal{D}).$$

Let $P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_{\mathcal{M}}(\mathcal{D})$, and let $\varrho \in \mathcal{R}_{\mathcal{M},\mathcal{D}}$ be the rule constructed according to Definition 11 for this fact.

By construction, each multichain pattern in each conjunction φ_r has cardinality at most $\texttt{count}_{\mathcal{D}}(\omega, c_{s_r}, c_u)$, so sufficiently many distinct paths exist in $\mathcal{D}$ to satisfy the body of ϱ. Therefore, the rule derives the same fact, and hence $P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_{\mathcal{R}_{\mathcal{M},\mathcal{D}}}(\mathcal{D})$.

Time Complexity. We measure data complexity, with the schema and model fixed. Let $\nu = \max_{1 \le k \le \delta} |P_k|$. Definition 11 constructs one rule for each predicted completion instance in $T_{\mathcal{M}}(\mathcal{D})$. Since each predicate has arity at most ν, the number of possible completed facts is

$$|T_{\mathcal{M}}(\mathcal{D})| = \mathcal{O}(\delta \nu \epsilon^{\nu}).$$

For each such fact, constructing the corresponding rule requires computing the realized path counts $\texttt{count}_{\mathcal{D}}(\omega, c_{s_r}, c_u)$ for all relevant path schemas ω and positions $r \neq t$. The number of path schemas is

$$\mathcal{O}(\delta^L \nu^{2L}),$$

and each path count can be computed in time $\mathcal{O}(\epsilon^L)$. Hence the cost of constructing one rule is

$$\mathcal{O}(\delta^L \nu^{2L} \epsilon^L).$$

Precomputing all bounds $\mathring{C}^{h,t}(\omega)$ contributes only lower-order overhead. Therefore, the total construction time is

$$\mathcal{O}\big(\delta^L \nu^{2L} \epsilon^{L+\nu}\big) = \mathcal{O}\big(\epsilon^{L+\nu}\big)\,.$$

□

B.4 Proof of Theorem 3

Theorem 3. *Let $\mathcal{M}$ be an MC-max model of depth L. Then, for every database $\mathcal{D}$, $T_{\mathcal{R}_{\mathcal{M}}^{\max}}(\mathcal{D}) = T_{\mathcal{M}}(\mathcal{D})$ when $\gamma = \beta$. Moreover, the program $\mathcal{R}_{\mathcal{M}}^{\max}$ contains at most $\mathcal{O}\big(\delta^{L+1} \cdot \nu^{2L+2}\big)$ rules, where $\nu = \max_{1 \le k \le \delta} |P_k|$.*

Proof. Using the matrices $\hat{\mathbf{M}}_d$ associated with the transitions indexed by $d \in \{1, \ldots, K\}$, Equation (5) can be rewritten as

$$\mathbf{v}_{\mathbf{c}}^{h,t}(u) = \max_{\substack{1 \le i \le N \\ 1 \le r \le |P_h|,\, r \neq t}} \left(\max_{\substack{(d_1,\ldots,d_L) \in \{1,\ldots,K\}^L \\ (\hat{\mathbf{M}}_{d_1} \otimes \cdots \otimes \hat{\mathbf{M}}_{d_L})(s_r,u)=1}} \prod_{j=1}^{L} \mathbf{b}^{h,t}(i,j,d_j) \right). \tag{13}$$

The constraint $(\hat{\mathbf{M}}_{d_1} \otimes \cdots \otimes \hat{\mathbf{M}}_{d_L})(s_r, u) = 1$ holds precisely when the path schema ω determined by $(d_1, \ldots, d_L)$ is realized from c_{s_r} to c_u in $\mathcal{D}$, since the identity matrix $\hat{\mathbf{M}}_K$ has no effect under max-product.

Soundness. We first prove soundness, namely that $T_{\mathcal{R}_{\mathcal{M}}^{\max}}(\mathcal{D}) \subseteq T_{\mathcal{M}}(\mathcal{D})$ for each dataset $\mathcal{D}$. Let

$$P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_{\mathcal{R}_{\mathcal{M}}^{\max}}(\mathcal{D})\,.$$

Then some rule in $\mathcal{R}_{\mathcal{M}}^{\max}$ with chain pattern corresponding to path schema ω and source position r is grounded in $\mathcal{D}$. Since the rule body is satisfied, ω is realized between c_{s_r} and c_u, so every sequence in S_{ω}^{L} satisfies the max-product constraint in Equation (13). Therefore,

$$\mathbf{v}_{\mathbf{c}}^{h,t}(u) \ge \texttt{wtm}_{\mathcal{M}}^{h,t}(\omega) > \gamma = \beta\,,$$

and the fact belongs to $T_{\mathcal{M}}(\mathcal{D})$.

Completeness. We show that $T_{\mathcal{M}}(\mathcal{D}) \subseteq T_{\mathcal{R}_{\mathcal{M}}^{\max}}(\mathcal{D})$ for each $\mathcal{D}$. Let

$$P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_{\mathcal{M}}(\mathcal{D})\,.$$

Then $\mathbf{v}_{\mathbf{c}}^{h,t}(u) > \beta$. By Equation (13), the maximum is attained at some $(d_1^*, \ldots, d_L^*)$, i^*, and r^*. Let ω^* be the path schema determined by this sequence. Then

$$\mathtt{wtm}_{\mathcal{M}}^{h,t}(\omega^*) \geq \prod_{j=1}^{L} \mathbf{b}^{h,t}(i^*, j, d_j^*) = \mathbf{v}_{\mathbf{c}}^{h,t}(u) > \beta = \gamma\,,$$

so the MC-max rule for (h, t, r^*, ω^*) belongs to $\mathcal{R}_{\mathcal{M}}^{\max}$ by Definition 9. The max-product constraint ensures ω^* is realized between $c_{s_{r^*}}$ and c_u, so the rule body is satisfied in $\mathcal{D}$. Therefore,

$$P_h(c_{s_1}, \ldots, c_{s_{t-1}}, c_u, c_{s_{t+1}}, \ldots, c_{s_{|P_h|}}) \in T_{\mathcal{R}_{\mathcal{M}}^{\max}}(\mathcal{D})\,.$$

Bound on rule number. Let $\nu = \max_{1 \leq k \leq \delta} |P_k|$. The number of path schemas in Ω is bounded by $|\Omega| = \mathcal{O}\big(\delta^L \cdot \nu^{2L}\big)$.

An MC-max rule is determined by a predicate P_h, a predicted position t, a source position $r \neq t$, and a path schema ω with $\mathtt{wtm}_{\mathcal{M}}^{h,t}(\omega) > \gamma$. For each (h, t, r) triple there are at most $|\Omega|$ choices of ω, so $|\mathcal{R}_{\mathcal{M}}^{\max}| = \mathcal{O}\big(\delta^{L+1} \cdot \nu^{2L+2}\big)$.

□

C Analytical Expression of $\mathtt{d}_{k,p,q}$

Given predicates $P_1, \ldots, P_\delta$ with arity $|P_i| \geq 2$ for $1 \leq i \leq \delta$, the position $\mathtt{d}_{k,p,q}$ of triple (k, p, q) is computed as

$$\mathtt{d}_{k,p,q} = \sum_{i=1}^{k-1} |P_i| \cdot (|P_i| - 1) + \begin{cases} (|P_k| - 1) \cdot (p-1) + q & \text{if } q < p \\ (|P_k| - 1) \cdot (p-1) + q - 1 & \text{if } q > p \end{cases}. \tag{14}$$

In turn, the triple (k, p, q) can be obtained for a given $\mathtt{d}$ satisfying $1 \leq \mathtt{d} \leq K - 1$ as

$$\begin{aligned} k &= \min \left\{ a \in \mathbb{Z}_{>0} \mid \sum_{i=1}^{a} |P_i| \cdot (|P_i| - 1) \geq \mathtt{d} \right\}, \\ p &= \min \left\{ b \in \mathbb{Z}_{>0} \mid \sum_{i=1}^{k-1} |P_i| \cdot (|P_i| - 1) + b \cdot (|P_k| - 1) \geq \mathtt{d} \right\}, \\ q' &= \mathtt{d} - \sum_{i=1}^{k-1} |P_i| \cdot (|P_i| - 1) - (p-1) \cdot (|P_k| - 1)\,, \quad q = \begin{cases} q' & \text{if } q' < p\,, \\ q' + 1 & \text{if } q' \geq p\,. \end{cases} \end{aligned} \tag{15}$$

D Comparison of Expressive Power of MC and MC-max

The following proposition shows that the expressive power of MC models is not equivalent to MC-max models, as MC models can capture rules that cannot be captured by MC-max.

Proposition 1. *There exists an MC model such that no MC-max model is equivalent to it, where equivalent means they produce the same result for each input dataset $\mathcal{D}$.*

Proof. We prove this proposition by giving an example of such MC model, and a specific input dataset $\mathcal{D}$. Then we show that no MC-max model is able to produce the same result as the MC model on this dataset.

We assume a signature consisting of a single predicate P with arity 4, and a set of constants $\{a_1, a_2, a_3, a_4, b, c, d, e, f, g, h\}$. The input dataset has an incomplete fact and three complete facts as $\mathcal{D} = \{P^4(a_1, b, c), P(a_2, b, c, d), P(a_3, b, e, f), P(a_4, g, c, h)\}$. Let the MC model $\mathcal{M} = (\mathbf{b}^{1,1}, \mathbf{b}^{1,2}, \mathbf{b}^{1,3}, \mathbf{b}^{1,4}, \beta)$ of rank $N = 1$ and depth $L = 1$, where $\beta = 0.9$, all elements of the tensors are 0 except $\mathbf{b}^{1,4}(1, 1, \mathtt{d}_{1,2,4}) = \mathbf{b}^{1,4}(1, 1, \mathtt{d}_{1,3,4}) = 0.5$. Following the computation process of the MC model as Equation (1), we are able to obtain the (only) constant d to replace the null value

in the incomplete fact. The completed fact is $P(a_1, b, c, d)$, which can be explained by the rule (16), containing two chain patterns of length 1.

$$P(x_1, x_2, x_3, y) \leftarrow P^4(x_1, x_2, x_3) \wedge P(u, x_2, v, y) \wedge P(w, z, x_3, y)\,. \tag{16}$$

Then we prove by contradiction that no MC-max model can produce the same result as $\mathcal{M}$ on $\mathcal{D}$. To this end, suppose there exists an MC-max model $\mathcal{M}'$ that yields the same result as $\mathcal{M}$ on $\mathcal{D}$. Then $\mathcal{M}'$ must predict the constant d for the incomplete fact, since d is predicted by $\mathcal{M}$. By definition of the MC-max model, it only utilizes one path with the highest weight from any existing constants (i.e., a_1, b, c in this case) from the incomplete fact to the target constant. We consider the following cases.

1. The model utilizes a path connecting a_1 and d in $\mathcal{D}$. As there is no such path in $\mathcal{D}$, this cannot be true.

2. The model utilizes a path connecting b and d in $\mathcal{D}$. In this case, we can find another path with the same path schema that connects b and f. In particular, by applying the mapping $\{a_2, a_4 \mapsto a_3; g \mapsto b; c \mapsto e; d, h \mapsto f\}$ to the path, i.e., replacing each constant with the one on the same position of predicate P as the fact $P(a_3, b, e, f)$. It is easy to see that the new path exists, and connects b and f. Besides, as the path schema is unchanged, the weight of the new path is the same as the one connecting b and d. Therefore, if the model $\mathcal{M}'$ derives d, it must also derive f in the result. This contradicts the fact that $\mathcal{M}$ only derives d as a result.

3. The model utilizes a path connecting c and d in $\mathcal{D}$. Analogous to case 2, we can find another path with the same path schema that connects c and h, with the mapping $\{a_2, a_3 \mapsto a_4; b \mapsto g; e \mapsto c; d, f \mapsto h\}$. Therefore, if the model $\mathcal{M}'$ derives d, it must also derive h in the result. This contradicts the fact that $\mathcal{M}$ only derives d as a result.

We have enumerated all possible cases for $\mathcal{M}'$ to derive d, and none of them satisfies the requirement that $\mathcal{M}'$ produces the same result as $\mathcal{M}$. This contradiction completes our proof that no such MC-max model exists that can produce the same result as $\mathcal{M}$ on $\mathcal{D}$. □

E Applying MC and MC-max to Complete Multiple Null Values

In this part, we explain that our models can be applied to incomplete facts with multiple null values as a direct extension. Figure 4 shows an example of this case, where some row(s) in the input tabular data contain more than one null value to be completed. As we introduced in Section 2, input tabular data can be viewed as facts with predicates.

Example 3. *Given the database as Figure 4, we have predicates* `ID-D-TG` *and* `D-PD-FG`*, complete facts* `ID-D-TG`(*1*, *Aspirin*, *COX-1*)*,* `ID-D-TG` (*2*, *Ibuprofen*, *COX-1*)*,* `D-PD-FG`(*Celecoxib*, *Colorectal-cancer*, *COX-2*)*, and an incomplete fact* `D-PD-FG`(*Aspirin*, ?, ?)*.*

ID	Drug	TargetGene
1	*Aspirin*	*COX-1*
2	*Ibuprofen*	*COX-1*
...	...	...

Drug	PreventDisease	FunctionalGene
Celecoxib	*Colorectal-cancer*	*COX-2*
...	...	...
Aspirin	**?**	**?**

Figure 4: An example of tabular cell completion with multiple null values in an incomplete fact, where each '?' denotes a missing value for the data cell.

Following the same definition of data predicates $P_1, \ldots, P_\delta$ and auxiliary predicates $\{P_i^t \mid 1 \leq t \leq |P_i|\}$ introduced in Section 4, we analogously represent the input tabular data as a dataset $\mathcal{D}$. Apart from keeping the same representation for facts with zero or one null value, we extend it for facts with multiple null values. Specifically, we introduce a special constant c_0 that is distinct from all the constants from the input tabular data, which intuitively denotes an 'unknown' constant. Then for each incomplete fact with predicate P_k and m null values on its $p_1, \ldots, p_m$ positions with $1 \leq k \leq \delta$ and $1 < m < |P_k|$, $\mathcal{D}$ contains m facts of the form $P_k^t(c_1, \ldots, c_{t-1}, c_{t+1}, \ldots, c_{|P_k|})$ for $t \in \{p_1, \ldots, p_m\}$, where each $c_j = c_0$ for $j \neq t$ and $j \in \{p_1, \ldots, p_m\}$.

Example 4. *In Figure 4, the incomplete fact* $\texttt{D-PD-FG}(\textit{Aspirin}, ?, ?)$ *is represented by two facts in* $\mathcal{D}$*, namely,* $\texttt{D-PD-FG}^2(\textit{Aspirin}, c_0)$ *and* $\texttt{D-PD-FG}^3(\textit{Aspirin}, c_0)$.

As the above formulation of dataset $\mathcal{D}$ has the same form as our input introduced in Section 3, we can apply our model as Definition 2 to $\mathcal{D}$. In particular, by additionally encoding the constant c_0 as $\mathbf{v}_{c_0}^{i,0} = \{0\}^\epsilon$, Equation (1) is well defined. Observe that, as c_0 is distinct from all constants in the input tabular data, introducing c_0 will not affect any of the adjacency matrices $\mathbf{M}_{p,q}^k$. Besides, as c_0 is initiated as a zero vector in Equation (1), the existence of c_0 has neither contribution nor effect on the final results of $\mathbf{v}_{\mathbf{c}}^{h,t}$.

Finally, to obtain complete facts by eliminating the existence of c_0, we post-process the $T_{\mathcal{M}}(\mathcal{D})$ (i.e., the result of applying model $\mathcal{M}$ to dataset $\mathcal{D}$) as follows. For each set of m facts $P_k(c_1, \ldots, c_{t-1}, c_t, c_{t+1}, \ldots, c_{|P_k|})$ in $T_{\mathcal{M}}(\mathcal{D})$ with a distinct $t \in \{p_1, \ldots, p_m\}$, satisfying $c_j = c_0$ for $j \neq t$ and $j \in \{p_1, \ldots, p_m\}$, and other constants c_i for $1 \leq i \leq |P_k|$ and $i \notin \{p_1, \ldots, p_m\}$ are the same for the m facts, we merge them into a single fact without c_0, by substituting all c_0 on the t-th position for $t \in \{p_1, \ldots, p_m\}$ with the (only) constant $c_t \in \{c_1, \ldots, c_\epsilon\}$ on the same position among these m facts.

Example 5. *In Figure 4, if the model produces* $T_{\mathcal{M}}(\mathcal{D}) = \{\texttt{D-PD-FG}(\textit{Aspirin}, \textit{Colorectal-cancer}, c_0), \texttt{D-PD-FG}(\textit{Aspirin}, c_0, \textit{COX-1})\}$*, the complete fact is obtained by merging the two facts as* $\texttt{D-PD-FG}(\textit{Aspirin}, \textit{Colorectal-cancer}, \textit{COX-1})$.

To apply the MC-max model for incomplete facts with multiple null values, recall that the MC-max model has the same form and input as the MC model (see Section 5); therefore, the above extension also applies to the MC-max model. Analogously, by adding the initializing vector of constant c_0 as $\mathbf{v}_{c_0}^{i,0} = \{0\}^\epsilon$, the transformation of MC-max model as Equation (5) is well defined. Then we are able to conduct the same post-process as above to eliminate c_0 and obtain the complete fact.

We exclude c_0 from the candidate domain for both model prediction and rule evaluation. The model and the extracted rules are applied independently to each missing position, after which the same deterministic merging procedure is applied to their outputs. Since c_0 occurs in no complete fact and is excluded from the candidate domain, its zero initialization does not change predictions for real candidates. Theorems 1 and 3 therefore give equal intermediate outputs, and applying the same deterministic merge preserves this equality. Therefore, the faithfulness of the resulting completed facts follows from Theorems 1 and 3.

F Experiment Settings

Additional details about the experiments are provided in this section, including the statistics of datasets, and the configurations used for training and evaluation.

F.1 Statistics of Datasets

For relational tabular datasets, we reused the three inductive datasets WP-IND, JF-IND, and MFB-IND from Yadati (2020). Each dataset consists of train, validation, and test sets, where the test set of each dataset contains the same predicates but also unseen constants w.r.t. the train and validation sets. Table 5 presents the statistics of each dataset. Compared with WP-IND and JF-IND, MFB-IND has a similar number of constants but significantly more facts, indicating a much higher density than the others. In addition to the number of facts, predicates, and the arity of each predicate, we also calculated the number of occurrences for each constant in the dataset. The constants in MFB-IND exhibited a higher median frequency of occurrence than those in the other datasets. This also suggests that MFB-IND has a denser structure with more paths between pairwise constants.

Following the standard evaluation protocol used in prior work (Yadati, 2020; Fatemi et al., 2020), we evaluated all methods on the same balanced sets of positive and negative candidate facts, ensuring controlled and directly comparable results. To use these datasets for tabular cell completion, each train set was randomly split into an incomplete set of facts $\mathcal{D}$ and a set of positive examples with a $3:1$ ratio. For each tabular dataset, the test set was similarly split at a $1:1$ ratio. The incomplete facts with a missing value were then obtained by masking one constant at each position in the positive examples. Besides, to evaluate model performance with negative examples, for each dataset

Table 5: Statistics of Tabular Datasets.

	#Train Facts	#Validation Facts	#Test Facts	#Constants	#Predicates	Arity		Constant Frequency
						median	max	(median)
WP-IND	4,139	1,138	1,139	4,463	32	3	4	1
JF-IND	6,167	659	645	4,785	31	3	4	2
MFB-IND	336,733	15,052	15,056	3,783	12	3	3	45

we randomly sampled negative facts by replacing the constant at each position of a positive example in the test set.

For binary datasets, we reused the 12 benchmark datasets for inductive knowledge graph completion (Teru et al., 2020). Table 6 presents the statistics of each dataset. Analogous to the tabular datasets, we computed the number of occurrences for each constant in the dataset. Compared with NELL-995 and WN18RR, FB15k-237 datasets in general have more predicates and a higher frequency of constants, indicating a greater diversity of incomplete facts and binary connections between pairwise constants. We reused the data splits in (Tena Cucala et al., 2022b; Wang et al., 2024a), including the incomplete set of facts, positive examples, and negative examples.

Table 6: Statistics of Binary datasets.

		#Train Facts	#Validation Facts	#Test Facts	#Predicates	Constant Frequency (median)
FB15k-237	V1	4,245	489	2,198	180	3
	V2	9,739	1,166	4,623	200	4
	V3	17,986	2,194	8,271	215	5
	V4	27,203	3,352	13,138	219	7
NELL-995	V1	4,687	414	933	14	2
	V2	8,219	922	5,062	88	2
	V3	16,393	1,851	8,857	142	2
	V4	7,546	876	7,804	76	2
WN18RR	V1	5,410	630	1,806	9	3
	V2	15,262	1,838	4,452	10	3
	V3	25,901	3,097	6,932	11	3
	V4	7,940	934	13,763	9	3

F.2 Model Configurations

For baseline models, we reused the original implementation of HyperGCN layers[1] from Yadati et al. (2019), and adapted it to the tabular cell completion task by adding an extra linear layer for binary classification to predict the correctness of an input fact. We used the standard version for better performance on WP-IND and JF-IND, while using the fast version on MFB-IND, as a single training epoch of the standard version on this dense dataset cost more than 24 hours.

For GMPNN models (Yadati, 2020), we reused their original implementation[2] for our task. We reused the default hyperparameter settings for all the baseline models. For the HC-MPNN model (Huang et al., 2025), we reused their original implementation[3], with the recommended settings and hyperparameters.

For LLM baselines, we implement Chain-of-Thought (Wei et al., 2022) with customized prompts and 3-shot, 5-shot examples, respectively. Each input contains an incomplete tabular data instance and a candidate fact, where the model is prompted to judge the validity of the candidate fact and output a binary answer. We accessed the GPT-5 mini model (OpenAI, 2025) via its official API. For TabLLM (Hegselmann et al., 2023), we followed the original implementation[4] and adapted it to our task. We further replaced its backbone LLM with a more advanced open-sourced model Llama-

[1] https://github.com/malllabiisc/HyperGCN

[2] https://github.com/naganandy/G-MPNN-R

[3] https://github.com/HxyScotthuang/HC-MPNN

[4] https://github.com/clinicalml/TabLLM/tree/main

3.1-8B-Instruct (Meta, 2024). We accessed the model from Hugging Face[5] and deployed it locally for fine-tuning and evaluation. The model was fine-tuned with QLoRA (Dettmers et al., 2023) for binary classification of the input candidate fact.

We implemented the MC and MC-max models based on Python 3.8[6] and PyTorch 2.0[7]. For training the MC and MC-max models, we applied the Adam optimizer with cross-entropy loss, and tuned the learning rate between 0.01 and 0.0001 for each model. Each model was trained for up to 10 epochs, and an early stopping strategy was employed when the validation loss increased during the training process. For MC and MC-max, we evaluated with the depth $2 \leq L \leq 3$ and the rank $1 \leq N \leq 3$. The threshold $\beta \in (0, 1)$ was tuned for each model to maximize the F1 score on validation sets. On binary datasets, we compared their best performance by tuning rank N between 1 and 3.

All non-LLM models are evaluated over five independent runs with distinct random seeds and fixed data splits, and we report the mean results across runs for each dataset. Results for the LLM baselines are obtained from a single run with temperature set to 0.

For rule extraction, we implemented the rule extraction based on Definition 11 for the MC model on all datasets used in our experiments. We also implemented the MC-max rule extraction based on Definition 9 for the MC-max models and set the rule extraction threshold $\gamma = \beta$.

All the experiments were performed on a workstation with an Intel Xeon E5-2670 CPU and an NVIDIA H100 GPU.

G Additional Evaluation Results

In this section, we provide further analyses of LLM performance, model training times, statistical significance, as well as database-specific rule extraction times for MC and example rules learned by MC-max.

G.1 Few-shot Prompting vs. Fine-tuning LLMs

Table 1 shows that both CoT and TabLLM exhibit imbalanced performance, particularly favoring conservative predictions and missing many valid completions. Although fine-tuning improves performance on some datasets, it does not consistently overcome this limitation.

In our task, evidence for a candidate for completion may be distributed across multiple tables and require several relational connections to be combined. A possible explanation is that serializing facts as text makes these connections difficult to identify and exploit consistently. Few-shot demonstrations may provide limited guidance for unfamiliar relational patterns, while fine-tuning could not ensure that the model generalizes these patterns beyond the training examples. Both approaches may therefore reject valid candidates when the supporting evidence is indirect. In contrast, our models explicitly aggregate evidence along relational paths, providing a structured mechanism for identifying such completions.

G.2 Training Time

Table 7 reports the training times for each model on WP-IND, JF-IND, and MFB-IND. Generally, the training time of all models ranges from several minutes to a few hours, depending on the dataset and model configuration. Among the baselines, HyperGCN trains faster than GMPNN on all datasets, while the three GMPNN variants exhibit similar training times, since their only difference is the strategy for aggregating neighborhood information.

All evaluated MC and MC-max configurations train faster than HyperGCN, the GMPNN variants, and TabLLM across all three datasets, and are also faster than HC-MPNN on JF-IND and MFB-IND. The advantage is particularly pronounced on MFB-IND, where MC and MC-max require 11.6–77.4 minutes, compared with 150.8–439.9 minutes for the baselines. For both MC and MC-max, the number of learnable parameters grows linearly with the rank, resulting in an approximately linear

[5] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

[6] https://www.python.org/downloads/release/python-3817/

[7] https://pytorch.org/get-started/pytorch-2.0/

growth in training time as the rank N increases. This indicates our models' flexibility to scale to larger datasets by adjusting the rank N as a hyperparameter. Overall, these results demonstrated the low training cost of MC and MC-max, suggesting their practical feasibility.

Table 7: Training Time (minutes) for Each Model.

	WP-IND	JF-IND	MFB-IND
HyperGCN	9.3	18.4	225.0
GMPNN-sum	21.7	24.5	439.9
GMPNN-mean	21.6	24.4	438.0
GMPNN-max	21.3	24.2	432.1
HC-MPNN	3.0	11.4	179.3
TabLLM (Llama 3.1)	95.6	110.3	150.8
MC ($L = 2, N = 1$)	2.3	3.0	29.0
MC ($L = 2, N = 2$)	4.1	5.6	39.2
MC ($L = 2, N = 3$)	6.2	8.2	46.2
MC-max ($L = 2, N = 1$)	1.7	2.3	11.6
MC-max ($L = 2, N = 2$)	3.3	4.2	38.1
MC-max ($L = 2, N = 3$)	4.7	6.3	50.1
MC-max ($L = 3, N = 1$)	2.7	3.7	25.9
MC-max ($L = 3, N = 2$)	5.5	7.0	67.8
MC-max ($L = 3, N = 3$)	7.7	10.3	77.4

G.3 Significance Test

Based on the results in Table 1, we use Welch's t-test to compare our models with non-LLM baselines on each dataset and a one-sample t-test against each single-run LLM result. Table 8 reports the resulting p-values, with an asterisk ($*$) indicating statistical significance at $p < 0.05$ and a dagger ($\dagger$) indicating that our model's mean score is lower than the baseline. Overall, the results demonstrate significant improvements over most baselines. Both MC and MC-max significantly outperform HyperGCN and all GMPNN variants in precision, accuracy, and PR-AUC across all datasets. Against HC-MPNN, both models achieve significantly higher precision on WP-IND and MFB-IND, while MC-max also achieves significantly higher accuracy on WP-IND. Compared with LLM baselines, both models achieve significantly higher recall across all datasets, and MC-max also achieves significantly higher F1 scores in every comparison.

G.4 Rule Extraction Time for MC models over a Specific Database

Table 9 reports the runtime of rule extraction for MC models on each dataset. Since the number of predicted facts obtained by applying $\mathcal{M}$ to $\mathcal{D}$ varies across datasets, we measured the average rule extraction time per 100 result facts obtained by applying $\mathcal{M}$ to $\mathcal{D}$, to enable a consistent comparison across datasets.

The average rule extraction time remained below 5 minutes per 100 predicted facts across all ranks. These results confirm the practical feasibility of our rule extraction approach. Additionally, we observed that the majority of the runtime was spent on computing the weights of every chain pattern and their cardinality upper bounds, while the time increment remained limited with growing size of the result sets.

G.5 Extended Analysis of Extracted Rules

Table 10 presents examples of top-weighted rules learned by MC-max from binary datasets. These rules are generally intuitive and easy to interpret for real-world users. They reflect meaningful relational schemas in the underlying knowledge graphs, such as the symmetric relation of `spouse`, transitive pattern of `also-see`, and subsumption relation of `subpart-of` and `location-located-within-location`.

Table 8: p-values of Significance Test for Results in Table 1. $*$: $p < 0.05$; †: lower than the baseline.

		MC ($L = 2, N = 3$)			MC-max ($L = 3, N = 3$)		
		WP-IND	JF-IND	MFB-IND	WP-IND	JF-IND	MFB-IND
HyperGCN	P	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	R	0.826†	0.209	0.415†	0.117	0.010*	0.041*
	Acc	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	PR-AUC	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	F1	0.001*	0.001*	<.001*	<.001*	<.001*	<.001*
GMPNN-sum	P	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	R	0.802†	0.009*†	0.377	<.001*	0.026*†	0.002*
	Acc	<.001*	0.003*	<.001*	<.001*	0.001*	<.001*
	PR-AUC	<.001*	0.001*	<.001*	<.001*	0.002*	<.001*
	F1	<.001*	0.386	<.001*	<.001*	0.063	<.001*
GMPNN-mean	P	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	R	<.001*†	<.001*†	0.794†	<.001*†	<.001*†	0.006*
	Acc	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	PR-AUC	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	F1	0.001*†	0.081†	<.001*	0.020*	0.049*	<.001*
GMPNN-max	P	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	R	<.001*†	0.002*†	0.001*†	<.001*†	0.003*†	0.026*†
	Acc	<.001*	<.001*	0.041*	<.001*	<.001*	<.001*
	PR-AUC	<.001*	<.001*	0.013*	<.001*	<.001*	0.015*
	F1	0.001*†	0.862	0.966†	<.001*	0.026*	0.001*
HC-MPNN	P	<.001*	0.714	<.001*	<.001*	0.241	0.001*
	R	<.001*†	0.077†	0.001*†	<.001*†	0.533†	0.028*†
	Acc	0.486	0.034*†	0.019*†	0.008*	0.470	0.651
	PR-AUC	0.003*†	0.058	0.800†	0.068	0.059†	0.512†
	F1	<.001*†	0.048*†	0.012*†	0.011*†	0.945†	0.584†
CoT (GPT-5 mini, 3-shot)	P	<.001*†	<.001*†	<.001*	0.003*†	<.001*†	0.001*
	R	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	Acc	<.001*	0.421†	<.001*	<.001*	0.026*	<.001*
	F1	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
CoT (GPT-5 mini, 5-shot)	P	<.001*†	<.001*†	<.001*	0.001*†	<.001*†	0.001*
	R	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
	Acc	<.001*	0.007*†	<.001*	<.001*	0.436	<.001*
	F1	<.001*	<.001*	<.001*	<.001*	<.001*	<.001*
TabLLM (Llama 3.1)	P	<.001*†	0.015*†	<.001*	0.002*†	0.053†	<.001*
	R	<.001*	0.019*	<.001*	<.001*	0.001*	<.001*
	Acc	<.001*	0.028*†	<.001*	<.001*	0.129	<.001*
	F1	<.001*	0.924†	<.001*	<.001*	0.004*	<.001*

Table 9: Rule Extraction Time for MC on Fixed Datasets (seconds per 100 facts).

		$N = 1$	$N = 2$	$N = 3$
WP-IND		11.89	14.61	15.70
JF-IND		11.34	13.78	14.48
MFB-IND		5.66	5.95	6.27
FB15k-237	V1	95.80	105.02	117.08
	V2	134.17	149.70	169.26
	V3	174.61	196.80	227.41
	V4	188.42	217.78	246.52
NELL-995	V1	0.17	0.20	0.24
	V2	10.71	12.59	14.30
	V3	47.18	57.08	64.25
	V4	6.97	8.45	9.42
WN18RR	V1	0.03	0.05	0.06
	V2	0.05	0.07	0.10
	V3	0.08	0.12	0.20
	V4	0.05	0.08	0.16

Table 10: Rules Learned by MC-max from Binary Datasets. (Auxiliary query atoms are omitted for readability.)

Binary Datasets	
FB15k-237	$\texttt{administrative-parent}(y, x) \leftarrow \texttt{contains}(x, y)$ $\texttt{director-film}(y, x) \leftarrow \texttt{film-edited-by}(x, y)$ $\texttt{language-spoken-in-country}(y, x) \leftarrow \texttt{official-language}(x, y)$ $\texttt{nominated-for-award}(x, y) \leftarrow \texttt{award-winner}(y, x)$ $\texttt{spouse}(y, x) \leftarrow \texttt{spouse}(x, y)$
NELL-995	$\texttt{company-economic-sector}(y, x) \leftarrow \texttt{subpart-of}(x, y)$ $\texttt{organization-hired-person}(y, x) \leftarrow \texttt{works-for}(x, y)$ $\texttt{subpart-of}(x, y) \leftarrow \texttt{location-located-within-location}(x, y)$ $\texttt{synonym-for}(y, x) \leftarrow \texttt{organization-also-known-as}(x, y)$ $\texttt{team-plays-against-team}(y, x) \leftarrow \texttt{team-plays-against-team}(x, y)$
WN18RR	$\texttt{also-see}(x, y) \leftarrow \texttt{also-see}(x, z) \wedge \texttt{also-see}(z, y)$ $\texttt{derivationally-related-form}(x, y) \leftarrow \texttt{derivationally-related-form}(y, x)$ $\texttt{derivationally-related-form}(y, x) \leftarrow \texttt{verb-group}(x, z) \wedge \texttt{verb-group}(z, y)$ $\texttt{has-part}(x, y) \leftarrow \texttt{hypernym}(z, x) \wedge \texttt{hypernym}(y, z)$ $\texttt{verb-group}(y, x) \leftarrow \texttt{verb-group}(x, y)$