
# Database Views as Explanations for Relational Deep Learning

Agapi Rissaki
Northeastern University, USA
rissaki.a@northeastern.edu

Ilias Fountalis
RelationalAI, USA
ilias.fountalis@relational.ai

Wolfgang Gatterbauer
Northeastern University, USA
w.gatterbauer@northeastern.edu

Benny Kimelfeld
Technion & RelationalAI, Israel
bennyk@cs.technion.ac.il

## ABSTRACT

In recent years, there has been significant progress in the development of deep learning models over relational databases, including architectures based on heterogeneous graph neural networks (hetero-GNNs) and heterogeneous graph transformers. In effect, such architectures state how the database records and links (e.g., foreign-key references) translate into a large, complex numerical expression, involving numerous learnable parameters. This complexity makes it hard to explain, in human-understandable terms, how a model uses the available data to arrive at a given prediction.

We present a novel framework for explaining machine-learning models over relational databases, where *explanations are view definitions* that highlight focused parts of the database that mostly contribute to the model's prediction. We establish such global abductive explanations by adapting the classic notion of determinacy by Nash, Segoufin, and Vianu (2010). In addition to tuning the tradeoff between determinacy and conciseness, the framework allows controlling the level of granularity by adopting different fragments of view definitions, such as ones highlighting whole columns, foreign keys between tables, relevant groups of tuples, and so on.

We investigate the realization of the framework in the case of hetero-GNNs, and develop a model-specific approach via the notion of learnable masks. For comparison, we propose model-agnostic heuristic baselines and show that our approach is both more efficient and achieves better explanation quality in most cases. Our extensive empirical evaluation on the RelBench collection across diverse domains and record-level tasks demonstrates both the usefulness of our explanations and the efficiency of their generation.

**PVLDB Reference Format:**
Agapi Rissaki, Ilias Fountalis, Wolfgang Gatterbauer, and Benny Kimelfeld. Database Views as Explanations for Relational Deep Learning. PVLDB, 19(7): 1643 - 1658, 2026.
doi:10.14778/3801059.3801075

**PVLDB Artifact Availability:**
The source code, data, and/or other artifacts have been made available at https://github.com/agapiR/rdl-explain.


Proceedings of the VLDB Endowment, Vol. 19, No. 7 ISSN 2150-8097.
doi:10.14778/3801059.3801075

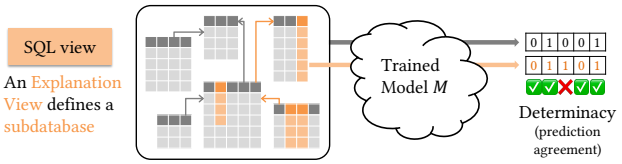



**Figure 1: An *explanation view* defines a subdatabase (or, more generally, a derived view). The rest of the database is randomly perturbed. As long as the view is preserved, a model gives similar predictions over the perturbed database as the original. We say the view *soft determines* the prediction.**

## 1 INTRODUCTION

Machine learning over relational databases was traditionally done via a reduction to unstructured data: flatten the data into a *feature matrix*, with a collection of predefined queries that produce features for every prediction instance, and then learn a flat model [10, 63, 93]. Progress in deep learning, and specifically Graph Neural Networks (GNNs) [121] and graph transformers [90] changed this practice: the best performing models now deploy prediction models directly from the structured data without ad-hoc feature extraction [40, 114].

Specifically, deep learning is commonly applied to relational databases by translating them into a graph representation: nodes represent entities (usually tuples), and edges represent relationships (usually foreign-primary key constraints). Graph-based deep models are then trained on these graphs [17, 29, 40, 112, 114]. To handle the varying node and edge *types* determined by the database schema, *heterogeneous* graph models are used [103, 116, 131]. This creates a two-paradigm pipeline: the relational database defines the graph, and the graph-based ML library learns the model. At inference, new data is exported to the graph representation, predictions are computed, and results are reintegrated into the database.

The deep model induced by the above graph ML approach over the database can be viewed as a complex numerical function constructed from the database's tuples and values, and parameterized by learned weights. Consequently, the model's behavior is inherently opaque and cannot be readily understood through direct inspection. To address this, the field of *Explainable Artificial Intelligence (XAI)* provides tools that shed light on different aspects of the model [5, 6, 31, 81]. These include *local explanations*, which account for the model's prediction on a specific instance, and *global explanations*, which aim to characterize the model's general behavior. Explanations may be *contrastive* (or counterfactual [13]), identifying hypothetical changes that would alter the model's outcome, or *abductive*, pointing to focused parts of the input that suffice to determine the outcome [81].

In this work, we study *global abductive explanations* for deep relational models. We focus on settings where the underlying schema consists of multiple interrelated tables with numerous attributes—a common real-world scenario. We aim to identify concise subsets of the data that best account for the model's predictions. Specifically, we address the question: *Which parts of the database does the model rely on most for its predictions?* Our goal is to provide global insights into the model's behavior by identifying the database fragments that are most influential across multiple predictions.

Example 1. *To illustrate the nature of our explanations, consider a relational database with schema* R(rid, $a_1$, ..., $a_5$), S(sid, rid, $b_1$, ..., $b_5$, flag), T(tid, rid, $c_1$, ..., $c_5$)*, where* rid *is the primary key of R and a foreign key in both S and T. All attributes $a_i, b_i, c_i$ are numerical, while* flag *is boolean. Consider two binary classification tasks predicting labels for R-records. The ground truth of Task* 1 *corresponds to whether at least* 3 *linked S-records have* flag = TRUE*, while Task* 2 *requires only at least* 3 *linked S-records (regardless of* flag *values). Task* 2 *thus depends purely on the join structure (*on R.rid = S.rid*), while Task* 1 *additionally depends on a specific attribute (i.e., the* flag*).*

*Now suppose a data scientist has trained a relational deep learning model that achieves high accuracy (ROC-AUC > 0.95) on both tasks. Applying our approach, the data scientist can understand what drives each model's predictions. For* Task 1*, explanations focused on important features reveal that only* S.flag *is critical, while all other feature columns ($a_i, b_i, c_i$) are excluded. To verify whether the relational structure also contributes, the data scientist requests relationship-focused explanations, which show that only the R-S join is significant (while the R-T join plays no role). Together, these findings show the model aggregates* flag *values over S-records linked to each R-record. Switching to* Task 2*, feature-focused explanations identify* no *feature column as important. How can the model predict accurately without features? Our data scientist now pivots to relationship-focused explanations and finds that the R-S join is important. This reveals that the model relies purely on connectivity structure (i.e., counting linked records) rather than attribute values.*

*This workflow, where users can select explanation types tailored to databases (columns, relationships, etc.) distinguishes our framework from conventional XAI approaches [15, 76, 97, 110]. In our example, traditional feature importance methods would fail to convey that Task 2 depends solely on connectivity structure. We implemented these tasks and confirmed our method produces these explanations.*

To gain insights such as those in Example 1, we propose a framework in which explanations are expressed as *view definitions* in SQL (or any query language of choice). This grounds explanations in a representation familiar to database users, in contrast to common alternatives that require machine learning expertise to interpret [19]. For graph ML, this would require knowledge of the graph modeling details or even the inner workings of GNNs [125].

To this aim, we deploy the classical notion of *determinacy*, introduced by Nash, Segoufin, and Vianu [91], that captures in a fundamental way when a collection of database views is enough to determine the result of a query. For reasons of usability, flexibility, and practical implementation, we propose a soft and statistical adaptation of this concept. Roughly speaking, the definition we propose identifies an explanation as *accurate* if the model is likely to have *similar predictions* on our database as it would have on alternative databases, as long as they agree on the views. Figure 1 shows a visual illustration. In addition to accuracy, we seek *concise* explanations that are easy to digest. Our framework naturally supports the tradeoff between accuracy and conciseness, allowing users to balance interpretability with fidelity to the model's behavior.

While our framework is *model-agnostic*, as it applies to any relational predictive model, efficient computation requires heuristics to avoid exhaustive search over the space of candidate explanation views, with *model-specific* algorithms offering further advantages by leveraging model structure. We instantiate the framework for the case of hetero-GNNs, for which we propose a model-specific approach for important classes of views: projections, joins over foreign keys, and selections (filters). The approach is based on learning *masked* variations of the GNN at hand, by applying masks on GNN components that correspond to the different view classes.

Lastly, we present an empirical evaluation of our techniques on the RelBench collection of databases and tuple-level tasks [98]. In particular, the experiments show the tradeoff between explanation accuracy and conciseness. As a side benefit, we show that, for most RelBench tasks, we can achieve models of similar quality to the ones publicly available, while using only a fraction of the database.

**Our main contributions.** (1) We introduce a model-agnostic framework for generating relational explanations of predictive models over relational databases, based on a statistical adaptation of determinacy. The framework uses relational queries, supports user-controlled granularity, and does not rely on internal model representations. (2) We instantiate the framework for hetero-GNNs, and develop an efficient model-specific approach via learnable masking that generates explanations without explicitly estimating soft determinacy. (3) We conduct extensive experiments on the RelBench benchmark across several databases and tasks. To facilitate comparison, we implement model-agnostic heuristic alternatives and demonstrate that our model-specific method consistently produces high-quality explanations at significantly lower computational cost.

## 2 PRELIMINARIES

In this section, we define the concepts and notation that we use throughout the paper.

### 2.1 Relational Database Concepts

We begin with database concepts and terminology.

**Databases.** A *relation schema* $\mathcal{R}$ is associated with a sequence $\text{att}(\mathcal{R}) = (A_1, \ldots, A_k)$ of distinct *attributes*, and a *key signature* $\text{key}(\mathcal{R}) = (B_1, \ldots, B_\ell)$, so that $\text{key}(\mathcal{R})$ is a subsequence of $\text{att}(\mathcal{R})$. Each attribute $A_i$ is associated with a (finite or infinite) *domain*, denoted $\text{dom}(A_i)$. If $\mathbf{C} = (C_1, \ldots, C_m)$ is a sequence of attributes of $R$, then we denote by $\text{dom}(\mathbf{C})$ the set $\text{dom}(C_1) \times \cdots \times \text{dom}(C_m)$. A *tuple* $t$ over a relation schema $\mathcal{R}$ is a member of $\text{dom}(\text{att}(R))$. For $i = 1, \ldots, k$, we denote by $t[A_i]$ the $i$-th value in $t$ (corresponding to the attribute $A_i$). If $\mathbf{C} = (C_1, \ldots, C_m)$ is a sequence of attributes from $\text{att}(\mathcal{R})$, then we denote by $t[\mathbf{C}]$ the tuple $(t[C_1], \ldots, t[C_m])$.

A *relation* over the relation schema $\mathcal{R}$ is a finite set of tuples over $\mathcal{R}$ so that $t_1[\text{key}(\mathcal{R})] \neq t_2[\text{key}(\mathcal{R})]$ for every two distinct tuples $t_1 \neq t_2$. If $r$ is a relation over $\mathcal{R}$ and $\mathbf{C} = (C_1, \ldots, C_m)$ is a

sequence of attributes from $\text{att}(\mathcal{R})$, then $r[\mathbf{C}]$ denotes the relation $\{t[\mathbf{C}] \mid t \in r\}$. We denote with $|r|$ the number of tuples in $r$.

A *database schema* $\mathcal{S}$ is associated with a set $\text{rel}(\mathcal{S})$ of *relation names*, and it maps every relation name $R \in \text{rel}(\mathcal{S})$ to a relation schema $\mathcal{S}(R)$. To ease the notation, we may write $\text{att}(R)$ and $\text{key}(R)$ as a shorthand notation for $\text{att}(\mathcal{S}(R))$ and $\text{key}(\mathcal{S}(R))$, respectively. In addition, a database schema $\mathcal{S}$ includes a set $\text{FK}(\mathcal{S})$ of *foreign-key constraints*, or *FK* for short, where an FK is an expression of the form $R[\mathbf{C}] \sqsubseteq \text{key}(S)$, where $R$ and $S$ are relation names in $\text{rel}(\mathcal{S})$, the sequence $\mathbf{C}$ consists of attributes from $\text{att}(R)$, the sequences $\mathbf{C}$ and $\text{key}(S)$ have the same length, and $\text{dom}(\mathbf{C}) \subseteq \text{dom}(\text{key}(S))$. A *database* $D$ over $\mathcal{S}$ maps every relation name $R \in \text{rel}(\mathcal{S})$ to a relation $D(R)$ over $\mathcal{S}(R)$, so that every key constraint and every constraint in $\text{FK}(\mathcal{S})$ is satisfied: the FK $R[\mathbf{C}] \sqsubseteq \text{key}(S)$ is *satisfied* if $D(R)[\mathbf{C}] \subseteq D(S)[\text{key}(S)]$.

**Relational queries and predictive models.** Let $\mathcal{S}$ be a database schema. A *query* $Q$ over $\mathcal{S}$ is associated with a relation schema, which we consistently denote by $\mathcal{R}_Q$, and it maps every database $D$ over $\mathcal{S}$ to a relation $Q(D)$ over $\mathcal{R}_Q$.

If $R \in \text{rel}(\mathcal{S})$ is a relation name, then an *$R$-model* is a query (possibly very complicated and with learned parameters) over $\mathcal{S}$ whose purpose is to make a prediction (e.g., classification or regression) for every tuple in $R$. Formally, an *$R$-model* (*of dimension $d$*) is a query $M$ with the following properties:

- *existence of prediction attribute* $A$: $\text{att}(\mathcal{R}_M) = (\text{key}(R), A)$, where $A$ is an attribute with $\text{dom}(A) = \mathbb{R}^d$, where $\mathbb{R}$ stands for the set of real numbers.
- *key signature agreement*: $\text{key}(\mathcal{R}_M) = \text{key}(R)$.
- *instance-prediction relation correspondence*: For every database $D$, $M(D)[\text{key}(R)] = D(R)[\text{key}(R)]$. In words, if $r = M(D)$ is the relation resulting from applying $M$ to $D$ , then $r$ has the exact same key values as the $R$-relation of $D$.

We refer to each tuple in $D(R)[\text{key}(R)]$ (notice the projection on only the key attributes) as an *instance* (to be classified by $M$).

Let $M$ be an $R$-model and $D$ a database, both over the schema $\mathcal{S}$. Let $s$ be an instance (i.e., a key tuple of $D(R)$) to which we apply the model. We denote by $M_D(s)$ the unique prediction of $M$ for $s$, that is, the value $t[A] \in \mathbb{R}^d$ of the unique tuple $t \in M(D)$ with $t[\text{key}(R)] = s$.

**Determinacy.** We briefly recall the concept of *determinacy* by Nash et al. [91]. Let $\mathcal{S}$ be a schema, let $V_1, \ldots, V_m$ be queries over $\mathcal{S}$, referred to as *views*, and let $Q$ be a query over $\mathcal{S}$. We say that $\{V_1, \ldots, V_m\}$ *determines* $Q$ if for all databases $D_1$ and $D_2$, if $V_i(D_1) = V_i(D_2)$ for every $i = 1, \ldots, m$, then $Q(D_1) = Q(D_2)$. Hence, knowing the result of every view suffices to determine the result of $Q$ without even looking at the database itself.

## 2.2 Heterogeneous Graphs

Next, we present the concepts and notation related to machine learning over heterogeneous graphs.

**Hetero-graphs.** A *heterogeneous featured graph*, or *hetero-graph* for short, is a quadruple $\mathcal{G} = (V, E, \tau, \mathbf{X})$ where:

- $V$ is a finite set of nodes;
- $E$ is a set of *typed directed edges*, that is, triples $(u, \lambda, v)$ where $u$ and $v$ are nodes in $V$ and $\lambda$ is an *edge type*;
- $\tau(\cdot)$ is a *node-type function*, mapping every node $v \in V$ to a type $\tau(v)$; and
- $\mathbf{X}$ is a *node-feature function* that maps every node $v \in V$ to a vector $\mathbf{X}_v \in \mathbb{R}^{d_{\tau(v)}}$ for some predefined *feature dimension* $d_{\tau(v)}$ for each node type.

Every edge type $\lambda$ in $G$ is associated with a *source type* $\text{src}(\lambda)$ and a *target type* $\text{tgt}(\lambda)$, and we require all edges $e = (u, \lambda, v)$ to be consistent in that $\tau(u) = \text{src}(\lambda)$ and $\tau(v) = \text{tgt}(\lambda)$. We denote by $\tau(V)$ the set $\{\tau(v) \mid v \in V\}$. If $\alpha \in \tau(V)$, then we denote by $V[\alpha]$ the set of nodes $v \in V$ with $\tau(v) = \alpha$.

We denote by $\mathcal{N}^{\text{in}}(v)$ (resp., $\mathcal{N}^{\text{out}}(v)$) the set of incoming (resp., outgoing) neighbors of $v$, that is, the set $\{u \mid \exists\lambda[(u, \lambda, v) \in E]\}$ (resp., $\{u \mid \exists\lambda[(v, \lambda, u) \in E]\}$). If $\lambda$ is an edge type, then we denote by $\mathcal{N}^{\text{in}}_\lambda(v)$ the set of nodes $u \in \mathcal{N}^{\text{in}}(v)$ such that $(u, \lambda, v) \in E$; analogously, we denote by $\mathcal{N}^{\text{out}}_\lambda(v)$ the set of nodes $u \in \mathcal{N}^{\text{out}}(v)$ with $(v, \lambda, u) \in E$.

**Node-centric models.** Let $\mathcal{G} = (V, E, \tau, \mathbf{X})$ be a hetero-graph, and let $\alpha \in \tau(V)$ be a type, which we refer to as a *prediction entity type*. A *$\alpha$-model* (of dimension $d$) is a function $M : V[\alpha] \to \mathbb{R}^d$, mapping every $\alpha$-node $v$ to a predicted vector $M(v)$. Note that the dimension $d$ depends on the task (e.g., it is one in binary classification and regression, and $k - 1$ in $k$-class classification), and is not necessarily the same as $d_\alpha$.

## 2.3 Heterogeneous Graph Neural Networks

A GNN defines a model over a graph by applying message passing to produce a vector representation of each node; in each step, called *layer*, the vector of a node is updated by aggregating the vectors of its neighbors, referred to as *messages*. A *hetero-GNN* operates over a hetero-graph similarly, except that messages of each edge type are aggregated separately. We give a more precise definition of a typical hetero-GNN next. We note that our definition is slightly more general and detailed than common definitions (e.g., [12]), stressing the role of labels and directionality in the GNN.

Let $\mathcal{G} = (V, E, \tau, \mathbf{X})$ be a hetero-graph and $\alpha \in \tau(V)$. A hetero-GNN with $L$ layers defines an $\alpha$-model using the functions $\mathbf{h}^{(\ell)}$ and $\mathcal{A}^{(\ell)}$, for $\ell = 0, \ldots, L$, constructed inductively as follows. First, define $\mathbf{h}^0(v) = \mathbf{X}_v$ for all $v \in V$. For $\ell = 0, \ldots, L-1$, define

$$\mathcal{A}^{(\ell)}(v, \lambda, o) = \text{AGG}(\{\!\{\mathbf{h}^{(\ell)}(u) \mid u \in \mathcal{N}^o_\lambda(v)\}\!\})$$

for every node $v$, edge label $\lambda$ and orientation $o \in \{\text{in}, \text{out}\}$ such that $\mathcal{N}^o_\lambda(v)$ is defined. Here, AGG is an aggregation function such as average or max, and $\{\!\{\cdot\}\!\}$ is the bag (multiset) notation. Next, for $\ell > 0$ we define

$$\begin{aligned}\mathbf{h}^{(\ell)}(v) = \text{UPD}(&\mathbf{h}^{(\ell-1)}(v),\\ &\mathcal{A}^{(\ell-1)}(v, \lambda_1, o_1), \ldots,\\ &\mathcal{A}^{(\ell-1)}(v, \lambda_q, o_q))\end{aligned} \tag{1}$$

for some ordering $(\lambda_1, o_1), \ldots, (\lambda_q, o_q)$ of all pairs $(\lambda, o)$ where $\mathcal{N}^o_\lambda(v)$ is defined. Here, UPD is an *update function*, which is realized as a neural model. A typical example of model architecture for hetero-GNNs [103] involves a linear combination followed by an activation function (e.g., ReLU). The model $M$ is given by $M(v) = \text{MLP}(\mathbf{h}^{(L)}(v))$ where MLP is a multilayer-perception function that transforms the last layer into the actual prediction. The learnable

parameters are incorporated in the functions AGG, UPD, and MLP. The same hetero-GNN defines an $\alpha$-model over any graph $\mathcal{G}'$ with the same node type set, edge type set and feature dimensions as $\mathcal{G}$.

## 2.4 Relational Learning via Heterogeneous Graphs

A common way to apply deep learning over relational databases is to convert the database $D$ into a hetero-graph $\mathcal{G}_D$, namely the *relational graph*, and then learn a hetero-GNN over $\mathcal{G}_D$ [18, 29, 40, 114]. We now describe this process more precisely for a typical conversion, specifically r2n (row-to-node) [114].

Let $\mathcal{S}$ be a schema, and let $D$ be a database over $\mathcal{S}$. The graph $\mathcal{G}_D = (V, E, \tau, \mathbf{X})$ is defined as follows. The set $V$ of nodes consists of the pairs $(R, t)$, where $R$ is a relation name and $t$ is tuple in $D(R)$:

$$V = \{(R, t) \mid R \in \mathsf{rel}(\mathcal{S}) \wedge t \in D(R)\}\,.$$

For a node $v = (R, t)$, we define $\tau(v) = R$, that is, the type of a node is the name of its relation. There is a directed edge $e = (v, \varphi, u)$ from a node $v = (R, t)$ to a node $s = (S, t')$ whenever the two are connected by a reference via an FK $\varphi$ of the form $R[\mathbf{C}] \sqsubseteq \mathsf{key}(S)$; in particular, the type of the edge is the FK that it corresponds to. Finally, the feature vector $\mathbf{X}$ is defined as follows. Let $R$ be a relation name with $\mathsf{att}(R) = (A_1, \ldots, A_k)$. For each attribute $A$ we assume a trainable encoder $\mathsf{Enc}_{R.A} : \mathsf{dom}(A) \to \mathbb{R}^{d_{R.A}}$ for some predefined dimension $d_{R.A}$. In addition, for each relation name $R$ we assume a trainable encoder $\mathsf{Enc}_R : \mathbb{R}^{d_{R_{enc}}} \to \mathbb{R}^{d_R}$, for some dimension $d_R$ and $d_{R_{enc}} = \sum_{i=1}^{k} d_{R.A_i}$. For a node $v = (R, t)$ we then define:

$$\mathbf{X}_v = \mathsf{Enc}_R\left(\mathsf{Enc}_{R.A_1}(t[A_1]) \oplus \cdots \oplus \mathsf{Enc}_{R.A_k}(t[A_k])\right) \qquad (2)$$

where $\oplus$ stands for vector concatenation. This formulation allows us to exclude some attributes $A$ of a relation $R$ from the encoding, simply by setting $\mathsf{Enc}_{R.A_i}$ to map every value to the empty vector. For example, in our implementation, we exclude key and foreign-key attributes from the features. Notice that this conversion process of $D$ into $\mathcal{G}_D$ ensures that the node type set, edge type set and feature dimensions of $\mathcal{G}_D$ depend solely on the schema $\mathcal{S}$.

Finally, let $R$ be a relation name in $\mathsf{rel}(\mathcal{S})$. An $R$-model $M'$ over $\mathcal{G}_D$ yields an $R$-model $M$ over the schema $\mathcal{S}$ in a straightforward way. For a tuple $t \in D(R)$:

$$M(t[\mathsf{key}(R)]) = M'(v)$$

where $v$ is the node $(R, t)$ in the node set $V[R]$.

# 3 RDL EXPLANATIONS VIA SOFT DETERMINACY

In this section, we introduce our framework, where we leverage the concept of determinacy to define RDL explanations.

## 3.1 Explanation Views

Let $\mathcal{S}$ be a schema with a relation name $R$, and let $M$ be an $R$-model. Let $D$ be a database over $\mathcal{S}$. We think of $M$ as a black-box complex model, involving a considerable number of computational components and learned parameters. By an *explanation* of $M$'s behavior on $D$, we refer to a collection of views $V$ (called *explanation views*), each indicating a focused component of $D$ that is important for $M$. Next, we define the explanation views more formally.

We assume an *explanation language* **EL**, which is simply a query language that we use as a formalism for defining explanation views. An explanation view $V$ states what component of the database $D$ (existing or derived) is relevant for $M$ to make the prediction $M_D(s)$ on any instance $s$; for that, $V$ can be parameterized by $s$, denoted $V\langle s\rangle$. This means that $V\langle s\rangle$ is an ordinary database query with the result $V\langle s\rangle(D)$, for every instance $s$. Recall that $s$ is represented as a key tuple in $D(R)$; hence, to represent explanation views, queries in **EL** can refer to the attributes of $\mathsf{key}(R)$ as constants. If $V$ ignores its parameter $s$, we say that it is *instance-agnostic*, otherwise it is *instance-specific*. Formally, $V$ is instance-agnostic if, for each database $D$ and instances $s$ and $s'$ it holds that $V\langle s\rangle(D) = V\langle s'\rangle(D)$. For the remainder of this work we consider instance-agnostic views.

Finally, by an *explanation* $E$ we mean a finite subset of concrete queries from **EL**. The quality of an explanation is measured using two criteria: *determinacy* (i.e., whether $E$ is enough to determine the behavior of $M$) and *conciseness* (i.e., how easy it is to grasp $E$).

## 3.2 Soft Determinacy

Consider an explanation $E$ of an $R$-model $M$ on a database $D$. Intuitively, $E$ "sufficiently determines" $M$ if $M$ behaves similarly when $D$ changes, as long as the result of $E$ is not affected by this change. Ideally, for any two databases that agree on the explanation views in $E$, the output of $M$ should be identical. Viewing $M$ as a query, this strict sufficiency notion is precisely *determinacy* (Section 2.1). To capture the imperfect nature of explanations, we relax in two dimensions. First, we do not require exact determinacy; instead, we account for *approximate agreement* by quantifying the distance between model predictions before and after the change. Second, we adopt a probabilistic adaptation, where we require determinacy *in expectation*, using our original database $D$ as reference and assuming a probability distribution of *contingency databases*. The two relaxations lead to an approximate probabilistic notion of determinacy, namely *soft determinacy*.

Formally, we assume a distribution $\Delta$ over contingency databases that agree on the schema $\mathcal{S}$ of $D$. To be comparable to $M(D)$, we require each database $D'$ in this distribution to agree with $D$ on the keys of $R$, that is: $[D'(R)[\mathsf{key}(R)] = D(R)[\mathsf{key}(R)]\,.$

To define approximate agreement, suppose that $M$ maps an instance $s$ to $a$, and an instance $s'$ to $a'$. We use a distance metric dist that quantifies how far $a$ is from $a'$ as $\mathrm{dist}(a, a')$. For example, in the special case of a binary classifier $M$ with predictions $\in \{0, 1\}$, it makes sense for dist to check for identity: $\mathrm{dist}(a, a') = \mathbf{1}_{\{a \neq a'\}}$.

Using the above, we arrive at the soft determinacy property for an explanation $E$, quantified by *deviation from determinacy*, so that lower deviation implies a better explanation.

Definition 2 (Deviation from Determinacy). *For an instance $s$, and a distribution $\Delta$ of contingency databases, the (instance-specific) deviation from determinacy $\mathsf{dev}_\Delta(E, s)$ of an explanation $E$ is the* expected distance *between the prediction for $s$ on $D$ and the prediction on a contingency $D'$, given that $D'$ respects the views:*

$$\mathsf{dev}_\Delta(E, s) := \mathop{\mathbb{E}}_{D' \sim \Delta}\left[\mathit{dist}(M_D(s), M_{D'}(s)) \;\middle|\; \bigwedge_{V \in E} V(D) = V(D')\right]$$

*Deviation from determinacy of $E$ is the empirical mean over instances:* $\mathsf{dev}_\Delta(E) := \frac{1}{|D(R)|} \sum_{s \in D(R)[\mathsf{key}(R)]} \mathsf{dev}_\Delta(E, s)$.

### 3.3 Explanation Conciseness

We associate with each $V \in \mathbf{EL}$ a *cost* related to its complexity. This cost can be syntactic, e.g., the number of symbols involved in the SQL representation. It can also be semantic, e.g., the number of tuples in the view $|V(D)|$. We denote this cost by $\text{cost}(V)$. We then define the cost of an explanation $E$ as $\text{cost}(E) = \sum_{V \in E} \text{cost}(V)$.

The computational problem of finding the best explanation can be phrased as an optimization problem:

Definition 3 (Soft determinacy explanations). *Given an R-model $M$ and a database $D$, the soft determinacy explanation problem asks for a solution $E \subseteq \mathbf{EL}$ to the following optimization objective:*

$$\min_{E} [\text{dev}_{\Delta}(E) + \lambda \cdot \text{cost}(E)],$$

*for a trade-off captured by $\lambda \in \mathbb{R}, \lambda \geq 0$.*

Clearly, there is a natural trade-off between achieving high soft determinacy (minimization of $\text{dev}_{\Delta}(E)$) and low explanation complexity (minimization of $\text{cost}(E)$). Consider the extreme case of $E$ recovering the entire database: $E$ fully determines $M$, but also fails to reduce the complexity of the model or the schema. On the other hand, a void explanation $E = \emptyset$ has minimal complexity, but likely does not capture any insight on the model's behavior.

The optimization problem in Definition 3 displays the challenges commonly associated with bilevel optimization [28, 33]. It involves two levels: the outer selects explanation views to minimize cost, while the inner estimates $\text{dev}_{\Delta}(E)$ by sampling and evaluating the model on multiple perturbed databases that respect the constraints imposed by the outer explanation selection. The outer level, i.e., searching among exponentially-many subsets of database components to find those that best preserve the model behavior, requires exploring a vast combinatorial space (see Example 4).

Example 4. *Consider the case where we seek explanation views that only involve projections over a single table with binary attributes, and we require exact determinacy ($\text{dev}_{\Delta}(E) = 0$). Suppose our model $M$ is a disjunction (OR) over all attributes, i.e., $M$ predicts 1 for all tuples except the all-zeros tuple $(0, \ldots, 0)$. A minimal explanation corresponds to a minimal set of columns such that every tuple has at least one 1 in those columns. This is precisely the classical hitting set problem, which is NP-hard. This example demonstrates that even for the simplest explanation language and model structure, finding optimal explanations requires considering exponentially many column subsets. The problem becomes more complex with more expressive explanations not restricted to projections.*

To address these challenges, in Section 4 we discuss several choices for realizing this objective into a more practical optimization problem. Additionally, in Section 5 we propose a learning-based solution for efficient explanation view discovery for hetero-GNNs.

## 4 INSTANTIATING THE FRAMEWORK FOR SELECTED SQL FRAGMENTS

The framework presented in Section 3 is abstract and requires the choice of an explanation language **EL** together with a concrete metric for conciseness and the distribution of contingency databases $\Delta$. In this section, we give specific instantiations that we deem useful and general. In the next sections, we propose heuristic implementations of these over GNN models and test them in experiments.

### 4.1 Explanation Languages (EL's)

Our explanation framework highlights the database components that provide the most useful information for the model. These may include important features, important feature combinations, or influential values and value ranges of particular features. In SQL terms, these explanations can be expressed through three core operations: projecting only the relevant attributes of relations, joining relations to reveal informative combinations, and selecting tuples that satisfy specific conditions. To capture these explanations, we focus on three simple SQL fragments utilizing the fundamental relational algebra operations: projections, joins, and selections. Throughout the remainder of this paper, we restrict our attention to instance-agnostic explanation views, for which we define suitable choices of **EL** and later propose efficient algorithms. We leave the study on instance-specific explanations for future work.

To define each **EL** precisely, we classify the attributes of a relation schema $\mathcal{R}$ in a database schema $\mathcal{S}$ into three categories. Given $\text{att}(\mathcal{R}) = (A_1, \ldots, A_n)$, an attribute $A_i$ is a *key attribute* if it appears in the key signature $\text{key}(\mathcal{R})$. It is a *foreign-key attribute* if it appears on the left-hand side of a foreign-key constraint in $\text{FK}(\mathcal{S})$. Any attribute that is neither a key nor a foreign key is called a *data attribute*. In the definitions of **EL** that follow, these three attribute types are treated separately.

(1) The language Projection can be used to highlight subsets of data attributes corresponding to important features, while always preserving all key attributes and foreign-key attributes. Formally, Projection contains views of the form:

$$\text{select key}(R), \mathbf{F}, A_1, \ldots, A_k \text{ from } R$$

which project onto a chosen subset of data attributes $A_1, \ldots, A_k$, while retaining the sequence $\text{key}(R)$ of key attributes and the sequence $\mathbf{F}$ of foreign-key attributes of $\mathcal{S}(R)$.

(2) The language FKJoin consists of foreign-key joins. Each explanation view in FKJoin is associated with a foreign-key constraint $R[\mathbf{C}] \sqsubseteq \text{key}(S)$ in the schema, and has the form:

$$\text{select} * \text{ from } R \text{ as } r, S \text{ as } s \text{ where } r[\mathbf{C}] = s[\text{key}(S)]$$

These views retain all tuples produced by joining $R$ and $S$ using foreign-key references. Note that more complex join patterns can be captured by a set of multiple views. Because FKJoin focuses on foreign keys, it always preserves all key attributes and data attributes of relations that appear in at least one explanation view.

(3) The language Selection focuses on filtering tuples based on attribute values. As with Projection, the emphasis is on data attributes, while key and foreign-key attributes are always preserved. Formally, Selection contains views of the form:

$$\text{select} * \text{ from } R \text{ where } \phi$$

where $\phi$ is a selection predicate that identifies an informative subset of tuples. For example, consider a database with a Product $P$ relation. The explanation view: "select $*$ from $P$ where $P.price < 50$ and $P.cat =$ 'Electronics'" highlights the group of affordable electronics.

**Cost model.** Explanation conciseness requires a precise definition of the cost model so that every explanation view $V \in E$ is

associated with a value cost$(V)$. Here we propose a simple cost model that focuses on minimizing the description cost of explanation views, making them easier for a user to understand. For Projection, cost$(V)$ is the number of highlighted data attributes, i.e., cost$(V) = k$. For FKJoin, each binary join incurs a cost of 1, i.e., cost$(V) = 1$. For Selection, cost$(V)$ depends on the complexity of the selection predicate $\phi$. Later, we will consider disjunctions of the form $\phi_1$ or ... or $\phi_l$, where each $\phi_i$ is an atomic predicate. In this case we simply assign cost$(V) = l$, i.e., the number of atomic predicates involved.

**Language composition.** We consider combinations of the three languages by constructing composite explanation views in the relational algebra sense. Importantly, these combination languages produce composite views rather than unions of separate explanation views belonging to one of the three defined languages. For composite views, the cost model is naturally extended: we sum the number of data attributes in projections, the number of joins, and the number of selection predicates involved.

## 4.2 Database Perturbations

Recall that the definition of deviation from determinacy (Definition 2) allows the liberty to choose any distribution $\Delta$ of contingency databases, as long as they agree with $D$ on the explanation views $E$.

Let us consider two extreme scenarios to illustrate the impact of $\Delta$. First, consider a $\Delta$ over databases $D'$ that differ from $D$ in just one value of a single tuple. A well-behaved model would likely give a similar output on $D$ and $D'$, irrespective of the chosen explanation $E$. Therefore, such $\Delta$ would not yield a discriminative criterion between candidate explanations. On the other hand, if $\Delta$ is the uniform distribution over all possible $D'$, no matter how different they are from $D$, it is not clear how one could obtain samples from $\Delta$. More importantly, the sampled $D'$ could be unrepresentative for the application domain of the original database $D$. Ideally, we would like to limit the explanatory power of $E$ to the useful regime, i.e., restricting $\Delta$ to *a local neighborhood* of $D$ that captures realistic variations while preserving relevance to the application domain.

Defining input perturbations is a fundamental challenge across ML explanation approaches that replace insignificant features with uninformative values. *Permutation Feature Importance* (PFI), a well-known model-agnostic feature selection technique [15, 88], replaces feature values by permuting them across instances, preserving marginal statistics while breaking correlations with the target. However, this approach may generate unrealistic data variations that violate feature correlations, thus forcing models to extrapolate [42, 57]. PFI variants address this by sampling from the conditional distribution $P(x_j|x_{-j})$ of the perturbed feature $x_j$ [32, 42, 133], but become computationally inefficient without strong assumptions like feature locality [133], assumptions more natural for images than tabular data. Subgroup-based methods partition instances into groups in order to reduce dependence before permuting [89], however they introduce additional complexity by training auxiliary models (e.g., decision trees) to identify appropriate subgroups for each feature.

We adapt these insights to relational databases by designing language-specific perturbations. For each explanation language **EL**, we provide strategies for realizing $\Delta$ by perturbing the original database $D$. Each strategy neutralizes information outside the explanation while respecting the explanation views. Following the PFI literature's concern with correlation preservation [42, 57, 89], we test both correlation-preserving and correlation-breaking variants in Section 6 and find results are mostly robust to this choice.

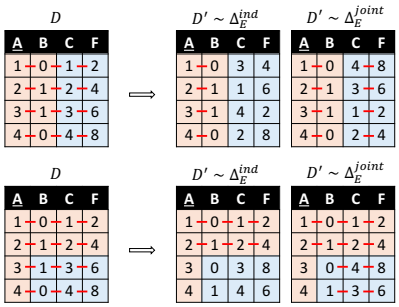



**Figure 2: Example permutations for Projection (upper) and Selection (lower). For Projection, only attributes $A$ and $B$ (in orange) are in Attr$(E)$ of explanation $E$. For Selection, the tuples (in orange) $\{(1, 0, 1, 2), (2, 1, 2, 4)\}$ are in Tups$(E)$. The key attribute $A$ is always retained.**

**Projection.** We permute columns not appearing in the explanation (independently or jointly), inspired by PFI.

Example 5. *The example in Figure 2 shows two column-wise permutation strategies. The attributes A and B are part of the explanation E, whereas C and F are not. Thus, we permute C and F to produce different $D'$. We can either apply the permutations to both attributes C and F jointly (left) or independently (right). Note that the latter completely breaks correlations between C and F, possibly creating unnatural tuples in $D'$. In this example, we have $F = 2 \cdot C$ for all tuples in D. Joint permutation preserves this property in $D'$.*

Formally, let Attr$(E)$ be the set of attributes used in a projection explanation $E$. Recall that Attr$(E)$ includes all key attributes. Because explanation views must be respected by $\Delta$, we only permute attributes not in Attr$(E)$. We define two permutation distributions that differ in how they handle correlations among non important attributes. The *joint column-wise permutation distribution* $\Delta_E^{\text{joint}}$ is the uniform distribution over all databases $D'$ where, for each relation in $D$, attributes $\notin$ Attr$(E)$ are permuted together by the same permutation. This preserves correlations among unimportant attributes while destroying correlations between important and unimportant attributes, and between the target and unimportant attributes. The *independent column-wise permutation* distribution $\Delta_E^{\text{ind}}$ is the uniform distribution over all databases $D'$ where each attribute $\notin$ Attr$(E)$ is permuted independently. This additionally breaks correlations among unimportant attributes.

**FKJoin.** The goal of a perturbation is to make joins uninformative unless they appear in $E$. Since the joins we consider always correspond foreign-key constraints, we apply a perturbation on the foreign key side. This perturbs the target join without affecting other joins on the side of the primary key.

Example 6. *Figure 3 (left) shows perturbations applied to the primary-foreign pair $T_2.B$ - $T_1.B$. One could consider permuting the foreign key $T_1.B$. But this strategy would not be effective in*

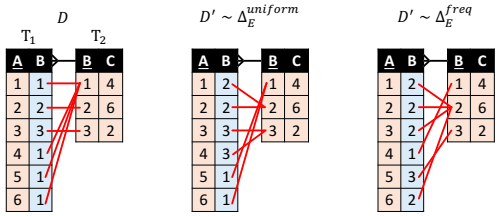



**Figure 3: Example 6: foreign key perturbations for FKJoin. Here, the foreign key $T_1.B$ is perturbed as it is not in $\mathsf{FK}(E)$.**

*breaking the predictive signal encoded in the actual join condition as it would retain the number of primary-foreign key pairs per unique value of $T_1.B$, i.e., 1 would appear 4 times while 2 and 3 would each appear once. In order to break the join information, we randomly replace each value of $T_1.B$ with a value in the domain $\{1, 2, 3\}$, either uniformly at random (right) or mimicking the underlying frequency distribution of foreign keys (middle). In the former each value in $\{1, 2, 3\}$ appears 2 times while in the latter 2 appears 4 times while 1 and 3 each appear once, preserving the frequencies 4, 1, 1.*

We consider two alternative strategies: In *uniform replacement* $\Delta_E^{\text{uniform}}$, each foreign key value is replaced independently and uniformly at random from its domain. In the *frequency-preserving replacement* $\Delta_E^{\text{freq}}$, the frequences of foreign key values in the original relation $r$ and the perturbed one $r'$ match: if a value occurs $i$ times in $r$, then there exists some value in $r'$ that occurs $i$ times, although not necessarily in the same tuples. Figure 3 shows an example, with $\mathsf{FK}(E)$ denoting the foreign keys in explanation $E$.

**Selection.** We adapt the column-wise permutation strategies used in the case of Projection for *all* data attributes, but applied only to the tuples that do not satisfy the selection condition of the explanation view. Figure 2 (lower) shows an example, with $\mathsf{Tups}(E)$ denoting the set of tuples that satisfy the selection condition.

## 5 GNN-SPECIFIC APPROACH

We now instantiate our framework (Section 3) for the case of GNNs. We present a model-specific approach that leverages properties of the model to efficiently solve the optimization in Definition 3: differentiability enables explanation discovery via gradient-based learning, while the model structure (i.e., feature encoding followed by message-passing) provides natural intervention points.

### 5.1 Mask Learning: From Discrete to Continuous Optimization

As discussed in Section 3.3, a direct solution to Definition 3 is not practically feasible. We now show how mask learning naturally arises as a continuous relaxation of the discrete soft determinacy problem, specifically for Projection explanations.

**Discrete Attribute Selection Problem.** Recall that soft determinacy (Definition 2) measures sufficiency of an explanation $E$ by evaluating the expected distance between $M_D(s)$ and $M_{D'}(s)$ over contingency databases $D'$ that respect the explanation views. For Projection explanations over $N$ data attributes, we seek a binary selection vector $\mathbf{a} \in \{0,1\}^N$ indicating which attributes to include in the explanation. To evaluate soft determinacy, we construct contingency databases by applying the joint column-wise permutation: for each tuple $t$, we replace the value of each attribute $A_i$ as:

$$t[A_i] \leftarrow a_i \cdot t[A_i] + (1 - a_i) \cdot t'[A_i]$$

where $t'$ is a randomly selected tuple from the same relation, and $a_i \in \{0,1\}$ determines whether attribute $A_i$ is fixed ($a_i = 1$) or replaced ($a_i = 0$). The soft determinacy objective then becomes:

$$\min_{\mathbf{a}\in\{0,1\}^N} \quad \mathbb{E}_s\, \mathbb{E}_\omega \left[\text{dist}(M_D(s), M_{\mathbf{a},\omega,D}(s))\right] + \lambda||\mathbf{a}||_0$$

where $s$ is an instance, $\omega$ represents the randomness in selecting replacement tuples $t'$, and $||\mathbf{a}||_0$ counts the number of selected attributes. However, optimizing over $\mathbf{a} \in \{0,1\}^N$ requires searching through $2^N$ combinations, which is intractable.

**Continuous Relaxation via Direct Masking.** A natural approach is to relax the binary constraint: replace $\mathbf{a} \in \{0,1\}^N$ with continuous mask values $\mathbf{m} \in [0,1]^N$. This yields the continuous optimization problem:

$$\min_{\mathbf{m}\in[0,1]^N} \quad \mathbb{E}_s\, \mathbb{E}_\omega \left[\text{loss}(M_D(s), M_{\mathbf{m},\omega,D}(s))\right] + \lambda||\mathbf{m}||_1$$

where we apply the masking operation at the feature encoding level. Concretely, let $\mathbf{x} \in \mathbb{R}^d$ be a feature vector for attribute $A_i$. Applying mask component $m_i \in [0,1]$ means replacing it with:

$$\mu_\omega(\mathbf{x}, m_i) = m_i \cdot \mathbf{x} + (1 - m_i) \cdot \mathbf{u}_\omega,$$

where $\mathbf{u}_\omega$ is the encoding of the replacement value from tuple $t'$, i.e., $\mathbf{u}_\omega = \text{Enc}_{R.A_i}(t'[A_i])$. Note that the $\ell_1$ penalty $||\mathbf{m}||_1$ serves as a convex relaxation of the $\ell_0$ cardinality constraint, encouraging sparsity [54]. Additionally, the function loss() is now a *continuous distance metric* (e.g., cross entropy, mean squared error).

This continuous relaxation enables gradient-based optimization using automatic differentiation frameworks. When $m_i = 0$, attribute $A_i$ is completely replaced, and as $m_i$ increases toward 1, the original value is progressively restored. In practice, we use stochastic optimization [64] as follows: for each optimization step we first sample a mini-batch of instances from $D(R)$ to compute the empirical loss, and then we apply one realization of the random replacement $\omega$ (e.g., one random permutation of tuples in each relation of $D$). After optimization, we threshold masks using $m_i \geq \delta$ to recover discrete explanations, a common technique for mask discretization [3].

Direct masking is also adopted for FKJoin and Selection as described in Section 5.2, providing a *surrogate optimization* for the soft determinacy objective.

**Alternative Approaches.** While direct masking has proven effective for interpretability in images [43] and graph neural networks [122], other techniques also exist for continuous relaxation of discrete selection problems. Policy gradient methods such as REINFORCE [119] treat selection as a stochastic policy, computing gradients via the likelihood ratio trick. Reparameterization approaches like Gumbel-Softmax [60, 79, 80] introduce auxiliary random variables to create differentiable approximations of discrete sampling. For example, L2X [21] uses these techniques to compute local explanations via feature selection with a fixed budget.

We adopt direct masking due to several practical advantages. First, it provides computational efficiency: requiring only a single forward pass per mask-learning step without sampling auxiliary

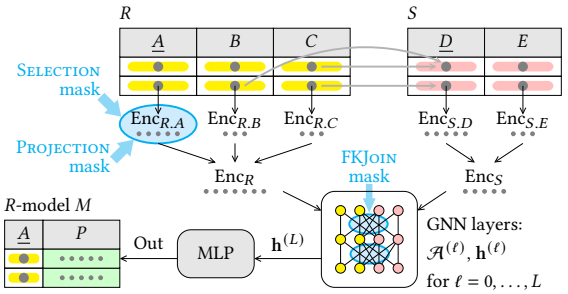



**Figure 4: RDL model pipeline with hetero-GNNs. The figure shows where different masks are located in the pipeline.**

variables, and producing stable gradients at all points in the continuous domain $[0, 1]$. Second, it naturally fits for the complex RDL model, where computation involves multiple stages (feature encoding, multi-layer message passing), allowing unified intervention across different explanation languages **EL** by modulating continuous internal representations.

**Quality of Mask Learning Solutions.** We discussed how mask learning arises as a natural continuous relaxation of the discrete soft determinacy problem for Projection. When the masks are discrete ($\mathbf{m} \in \{0,1\}^N$), the objectives are equivalent: $\text{dev}^{\text{joint}}_{\Delta_E}(E) = \mathbb{E}_s\, \mathbb{E}_\omega \left[\text{loss}(M_D(s), M_{\mathbf{m},\omega,D}(s))\right]$. In practice, we learn continuous masks and threshold them to obtain discrete explanations. We now argue that this approach yields high-quality solutions.

Consider the simplified setting of a binary classifier $M_C$ that relies on *precisely* $k$ data attributes. First, there exists a high-quality Projection explanation $E^*$ for $M_C$: since $M_C$ uses exactly $k$ columns, the $k$-column explanation selecting these achieves cost $\ell^* = \lambda \cdot k$ (zero task loss plus regularization). Second, the mask learning objective promotes this solution: any *smaller* mask (with $k' < k$ attributes) will incur substantial task loss since important features are masked, thus predictions flip with probability $\epsilon > 0$, yielding expected cost $\ell' = \epsilon + \lambda \cdot k' >> \ell^*$ for appropriate $\lambda$. Conversely, any *larger* mask (with $k'' > k$ attributes) incurs higher regularization cost $\lambda \cdot k'' > \ell^*$ without reducing task loss. Hence, the mask learning objective promotes the optimal $k$-column explanation $E^*$.

## 5.2 Language-Specific Masking

Depending on **EL**, masking is applied at the feature encoding stage or the message-passing stage of the model, as depicted in Figure 4.

**(1) Projection.** We apply mask components attribute-wise in the feature encoding space. Since Projection necessarily includes all primary and foreign keys, we restrict masks to data attributes.

For each relation $R \in \text{rel}(\mathcal{S})$, we assign one mask component $m_{R.A_i} \in [0,1]$ per data attribute $A_i$, $i = 1, \ldots, n$. For each node $v = (R, t)$, the same mask vector $(m_{R.A_1}, \ldots, m_{R.A_n})$ is applied at feature encoding, changing Equation (2) as follows:

$$\mathbf{X}_v = \text{Enc}_R(\mu(\text{Enc}_{R.A_1}(t[A_1]), m_{R.A_1}) \oplus \cdots \oplus \mu(\text{Enc}_{R.A_n}(t[A_n]), m_{R.A_n}))$$

Recall that $\mu(\mathbf{x}, m)$ applies the mask component $m$ to the encoding vector $\mathbf{x}$. Every time we apply masking attribute-wise, we first consider a random permutation to the tuples of $D(R)$, which assigns for each tuple $t$ a replacement tuple $t'$. Then to apply masking to $v = (R, t)$ for each attribute $A_i$ we use the replacement vector $\mathbf{u}_i = \text{Enc}_{R.A_i}(t'[A_i])$, i.e., the encoding of the value of $A_i$ in $t'$. This ensures the replacement of the $t[A_i]$ encoding is uninformative, yet realistic. Masks for all relations and attributes are learned jointly.

**(2) FKJoin.** We apply masks at message-passing, learning a mask value $m_{\lambda_i} \in [0,1]$ for every edge type $\lambda_i$. Since every foreign-key pair corresponds to an edge type (see Section 2.4), $m_{\lambda_i}$ indicates the importance of the corresponding join. Equation (1) becomes:

$$\mathbf{h}^{(\ell)}(v) = \text{UPD}(\mathbf{h}^{(\ell-1)}(v), \mu(\mathcal{A}^{(\ell-1)}(v, \lambda_1, o_1), m_{\lambda_1}), \ldots, \mu(\mathcal{A}^{(\ell-1)}(v, \lambda_q, o_q), m_{\lambda_q}))$$

For each edge type $\lambda_i$, the same mask component $m_{\lambda_i}$ is applied at the aggregate vector of messages irrespective of the direction $o_i$, as both directions correspond to the same join. Also, the same mask components are used across all layers $\ell$, because message-passing corresponds to the same join operation irrespective of the GNN layer. We replace each aggregate with a zero replacement vector $\mathbf{u}$.

**(3) Selection.** First, consider the case of a single selection predicate $\phi$. We assign a mask $m_\phi \in [0,1]$ to all tuples that satisfy $\phi$ and a mask $m_{-\phi} \in [0,1]$ to the rest. For each node $v = (R, t)$, we set $m_{(R,t)} = m_\phi$ if $\phi(t) = 1$ and $m_{(R,t)} = m_{-\phi}$ otherwise. Masking is then applied per tuple by adapting Equation (2) into:

$$\mathbf{X}_v = \mu(\text{Enc}_R\left(\text{Enc}_{R.A_1}(t[A_1]) \oplus \cdots \oplus \text{Enc}_{R.A_n}(t[A_n])\right), m_{(R,t)})$$

Similarly to Projection, when masking $v = (R, t)$, we replace $\mathbf{X}_v$ with $\mathbf{u} = \mathbf{X}_{v'}$ for a replacement $v' = (R, t')$, $t' \in D(R)$ according to a random permutation of tuples in $D(R)$.

For the implementation of Selection, we consider conditions that involve disjunctions of atomic predicates $\phi_1 \vee \cdots \vee \phi_l$. Following real-valued logic techniques that map logical formulas into differentiable form [69], we employ the Łukasiewicz t-conorm for disjunctions. Thus, the mask values per tuple are calculated as $m_{(R,t)} = \min\{1, \sum_{\phi_i(t,\mathbf{c}_i)=1} m_{\phi_i}\}$. A similar strategy may be applied for conjunctions. In our experiments (Section 6), we restrict ourselves to categorical attributes with equality predicates and numerical attributes with range predicates defined via quantiles.

In all cases, continuous mask values are mapped into discrete explanations using a threshold $\delta > 0$; $m'_x = 1$ if $m_x \geq \delta$ and 0 otherwise. Components $x$ with $m'_x = 1$ are included in the explanations; the rest are omitted.

# 6 EXPERIMENTAL EVALUATION

**Metrics.** The main metric is deviation from determinacy $\text{dev}_\Delta(E)$ (see Definition 2). The distance dist is absolute difference of probabilities for binary classification and normalized absolute difference $\text{dist}(x, y) = \frac{|x-y|}{|x|+|y|+\epsilon}$ for regression, both bounded in $[0,1]$. Each $\text{dev}_\Delta(E)$ estimate is over 5 perturbation samples; we report both mean and standard deviation. Second, we evaluate the explanation size $k$. Recall that this is the number of projected data attributes for Projection, the number of join conditions for FKJoin, and the number of predicates for Selection. Finally, we report the generation time, which corresponds to training time for mask learning.

**Data.** We use RelBench, the standard benchmark for RDL [98] with real-world databases and diverse predictive tasks. We focus on node-level tasks, i.e., regression and binary classification. Dataset

details are provided in the online appendix. As Table 2 shows, we denote each combination of database $i$ and task $j$ as D$i$T$j$.

**Explained Models.** We train RDL models from scratch. For each database, we construct a graph (Section 2.4) and use PyTorch Frame [58] for feature encoders. The GNN is implemented in PyG [41] with 32-dimensional channels. We perform hyperparameter tuning over the learning rate, batch size, and number of GNN layers, and use the train/val/test split of the benchmark. Our models reach performance similar to the one previously reported [98] for GNN approaches. For each explanation task, we sample 100 instances for mask training and 100 instances for evaluation. For classification tasks, we report a class-balanced $dev_\Delta$ that averages over an equal number of instances from each class, avoiding majority-class dominance. For temporal tasks we perform temporal sampling [40] to ensure time-consistency.

**Methods.** (1) Mask-based: Our method as introduced in Section 5, with early stopping and thresholding with $\delta = 0.1$. Note that, on average, *only* 1.97% of mask values for Projection and 2.17% for FKJoin fall in the range $[0.05, 0.2]$ and are sensitive to small $\delta$ variations. (1a) `Column Mask`: Masks for Projection. (1b) `FKPK Mask`: Masks for FKJoin. (1c) `Filter Mask`: Masks for Selection.

Since no prior work generates SQL explanations for RDL, we implement a set of baselines. (2) Ranking-based for Projection: We order data attributes according to their importance, and select the top-$k$ for each target explanation size $k$. A known limitation of such methods is that they ignore dependencies between attributes [88]. (2a) `Local Impact`: The importance of a data attribute is computed as $dev_\Delta$ when the attribute is taken as an explanation by itself. The lower the $dev_\Delta$ the more important the attribute. (2b) `PFI`: We adapt Permutation Feature Importance (PFI) [88] by measuring the change in $dev_\Delta$ when each attribute is removed. Attributes are ranked by impact, where higher deviation implies higher importance.

(3) Greedy approach for Projection and FKJoin: (3a) `Greedy`: Starting from the empty set, we iteratively add the attribute that yields the largest reduction in $dev_\Delta$. (3b) `Greedy Expansion`: At each step, we add a join that maximally reduces $dev_\Delta$ while ensuring connectivity to the prediction entity. This is more computationally feasible than `Greedy` due to fewer join candidates at each iteration.

(4) `Random Subset` for Projection: We randomly sample the data attributes and report the average over 5 samples.

**Experiment details.** For text attribute embeddings we use a BERT-based model[1]. Experiments are done on a VM with an RTX 6000 Ada GPU, 14 vCPUs, and 188 GB of RAM.

**Evaluation plan.** We evaluate mask learning separately for Projection and FKJoin. For fair comparison, we use the same explanation size for all methods, denoted as $k^*$, which we determine by mask thresholding as discussed in Section 5. We also assess the robustness of $dev_\Delta$ under different database perturbation strategies $\Delta$. Finally, we conduct a case study to show end-user usefulness.

## 6.1 Projection Evaluation

Figure 5 reports the results for Projection across all datasets and tasks, including averages (AVG).Figure 5a and Figure 5b show $\text{dev}_\Delta$ for two different perturbation strategies. The main takeaway is that `Column Mask` produces higher quality explanations (in terms of $\text{dev}_\Delta$) than the baselines on most tasks, and on average, while outperforming them on execution time (Figure 5c) by 1-2 orders of magnitude for large databases. This is expected since `Column Mask`'s running time is dependent on the size of the model and the size of the explanation task, which is usually small. Among the baselines, `PFI` gives the best results both on quality and on time. However, all baselines suffer from their costly dependence on the schema size, i.e., the number of data attributes that they consider. On database D1 `PFI` is faster, but this is due to the fact that the database is small and has a concise schema. Comparing the two perturbation strategies, they appear to yield similar results, indicating that $\text{dev}_\Delta$ is robust to the choice of permutation approach.

While Figure 5 assumes the same explanation size $k^*$, Figure 6 shows $dev_\Delta$ for various explanation sizes $k$, illustrating the tradeoff between the two. As the size $k$ increases, we generally observe $dev_\Delta$ decreasing, i.e., improving explanation quality. This is not true for `Random Subset`, which shows that the choice of attributes in the explanation is crucial. For smaller $k$ than $k^*$, the baselines are often better than `Column Mask`, but `Column Mask` usually outperforms them for the optimal $k^*$ (i.e., the end of the axis).

## 6.2 FKJoin Evaluation

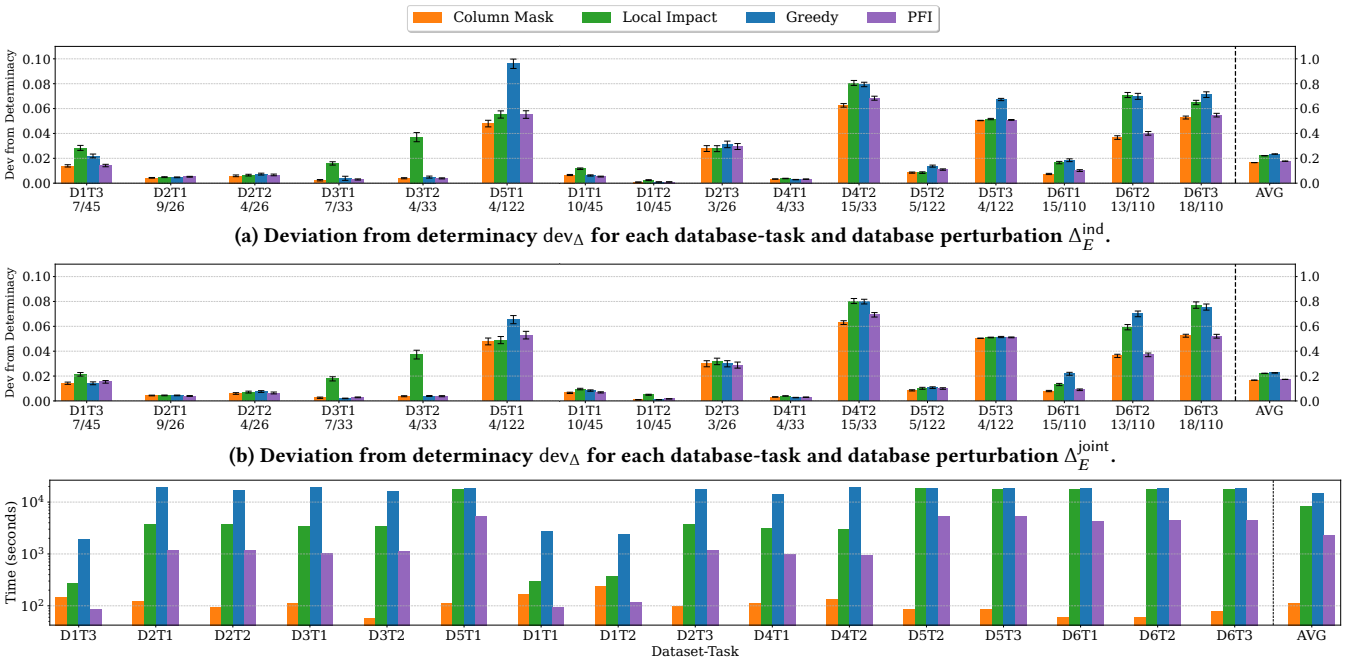
Our method `FKPK Mask` achieves an average (across all datasets and tasks) $\text{dev}_\Delta$ of $0.1366 \pm 0.0020$ for $\Delta_E^{\text{uniform}}$ and $0.1359 \pm 0.0022$ for $\Delta_E^{\text{freq}}$. The baseline `Greedy Expansion` yields $0.1400 \pm 0.0022$ and $0.1415 \pm 0.0023$ for the respective distributions. In terms of $\text{dev}_\Delta$, the gap between `FKPK Mask` and `Greedy Expansion` is smaller than in previous experiments, but `FKPK Mask` still provides a small average advantage, winning 54% of the time (15 out of 28 comparisons, excluding datasets with $\leq 2$ joins). As before, the mask-based approach `FKPK Mask` is significantly faster than the greedy baseline `Greedy Expansion`, requiring only 101.46 seconds on average compared to 2114.02 seconds under $\Delta_E^{\text{uniform}}$ and 2844.50 seconds under $\Delta_E^{\text{freq}}$, offering a speedup of 1-2 orders of magnitude consistent across the vast majority of datasets-tasks. The differences between the two perturbation strategies are again minor verifying that our metric is robust across the perturbation variants considered.

## 6.3 Case Study: Diagnosing Model Behavior

We demonstrate how our framework's expressivity enables users to diagnose model behavior through a series of controlled scenarios on the rel-trial dataset [27], a database of clinical trial reports. It contains 15 tables with a total of 5,852,157 rows and 140 attributes, out of which 110 are data attributes and 30 are key or foreign-key attributes. We focus on the binary classification study-outcome (D6T1), that predicts whether a trial achieves its primary outcome. To show how different **EL**s reveal distinct issues, we also construct 3 synthetic variants of the original task, summarized in Table 1.

**Original Task.** Our method identifies the deciding factors for whether a study will achieve its outcome. It highlights 15 data attributes and 10 foreign-key pairs using Projection and FKJoin respectively. Utilized facilities ("studies JOIN facilities_studies JOIN facilities") and study designs ("studies JOIN designs") are revealed as key predictors. The relation designs originally contains

[1] https://huggingface.co/sentence-transformers/distilbert-base-nli-mean-tokens, Accessed: 2025.

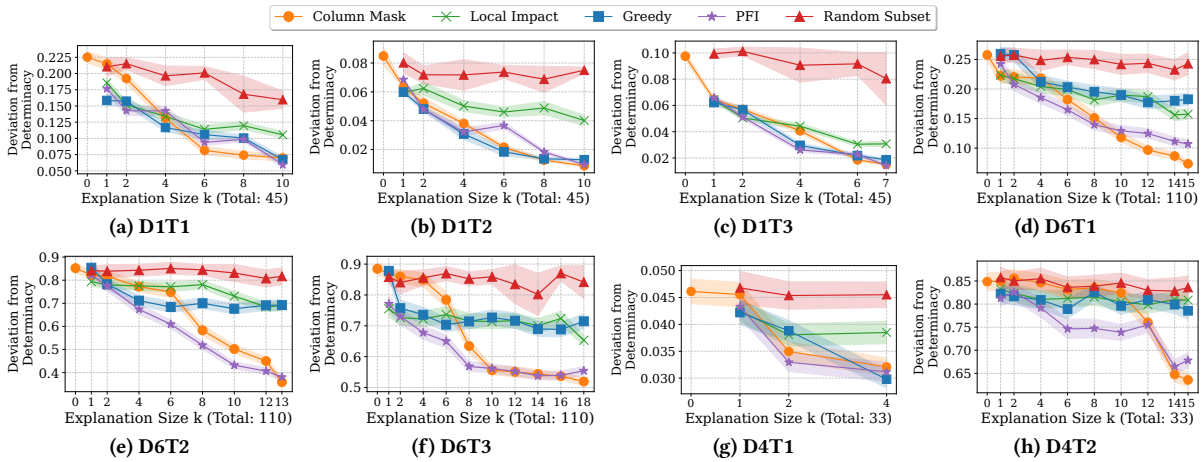



(a) Deviation from determinacy $\text{dev}_\Delta$ for each database-task and database perturbation $\Delta_E^{\text{ind}}$.

(b) Deviation from determinacy $\text{dev}_\Delta$ for each database-task and database perturbation $\Delta_E^{\text{joint}}$.

(c) Time to obtain explanation for each database-task and $\Delta_E^{\text{ind}}$. Results for $\Delta_E^{\text{joint}}$ are similar.

**Figure 5: Evaluation for PROJECTION. For visualization purposes, tasks are separated into "easy" (left) and "hard" (right) ones, with separate $\text{dev}_\Delta$ scales. The average (AVG) across all tasks is also shown. Below the dataset and task name, we indicate the explanation size $k^*$ and the total number of data attributes.**




(a) D1T1 (b) D1T2 (c) D1T3 (d) D6T1

(e) D6T2 (f) D6T3 (g) D4T1 (h) D4T2

**Figure 6: Soft determinacy versus conciseness for PROJECTION explanations, for 8 tasks and database perturbation $\Delta_E^{\text{ind}}$.**

13 data attributes, of which 3 are important: allocation, intervention_model, primary_purpose. The relation facilities contains 3 additional important data attributes: name, city, and country, indicating that names and locations of utilized facilities highly influence the study success or failure. Notably, *no* attributes from the outcomes relation (e.g., outcome type or description) are included, indicating they do not help predict outcome achievement. We further examine the important selection predicates discovered by our method using SELECTION. Our method detects the following predictive predicates: "designs.allocation = 0" selects single-arm trials, "designs.intervention_model = 0" selects single-group interventions and "studies.phase $\in \{0, 2, 4\}$" selects trials based on the phase.

**Scenario 1: Detecting Column-Level Data Leakage.** We simulate a realistic mistake: a user inadvertently includes a prelim_evaluation table where studies receive an evaluation from multiple reviewers (many-to-many join) and the numerical column rating ($\in \{1, 2, 3, 4, 5\}$) as part of the evaluation record is perfectly correlated with outcome (high grades ($\in \{4, 5\}$) $\rightarrow$ positive outcomes, low grades ($\in \{1, 2\}$) $\rightarrow$ negative outcomes). The trained model achieves suspiciously high performance (ROC-AUC $\simeq$ 100%).

**Table 1: Summary of case study scenarios showing how different explanation languages reveal distinct patterns.**

| Scenario | EL | Size | Key Artifacts |
|---|---|---|---|
| Original | ALL | 30 | facilities, designs, etc. |
| Column Leakage | PROJECTION | 1 | SELECT rating FROM prelim_evaluation |
| Tuple Leakage | SELECTION | 3 | eval_category IN ("A", "B", "C") |
| Structural | FKJOIN | 1 | studies JOIN sponsors_studies |
| | PROJECTION | 0 | No sponsor data attributes |

Using PROJECTION, our method immediately flags the rating column as highly important ($\text{mask}_{\text{rating}} \simeq 1$), while mask values for all other columns are $\simeq 0$. This provides a clear signal to investigate potential leakage in this specific column.

**Scenario 2: Pinpointing Tuple-Level Leakage.** We now make the leakage more subtle: each evaluation record refers to a certain evaluation aspect (e.g., "financial", "ethics", "methodology" etc.) indicated by a column eval_category that admits 10 categorical values ("A", ..., "J"). In this case, only ratings on the first three categories ("A", "B", "C") are correlated with outcomes (as before) while the rest are uncorrelated. Model performance is again high. PROJECTION identifies rating as important along with eval_category. Subsequently, to understand *which* evaluations cause leakage, we use SELECTION on the eval_category column. The method precisely identifies the predicate eval_category IN ("A", "B", "C") with high mask value, while other evaluation records have low importance. This fine-grained diagnosis enables targeted data cleaning.

**Scenario 3: Understanding Structural Importance.** We make the outcome depend purely on structure: outcome = 1 if the study has multiple sponsors, else 0. The model learns this pattern well. FKJOIN explanations reveal the sponsors_studies join as critical. However, PROJECTION explanations show that *no specific sponsor attributes* are important, the predictive signal comes entirely from the join structure (i.e., counting sponsors). This illustrates that expressivity across multiple languages is essential: neither PROJECTION nor FKJOIN alone would provide the complete picture.

## 6.4 Retraining Results

As a sanity check, we retrain the models using only the subset of the data retained by our PROJECTION and FKJOIN explanations. As Table 2 shows, performance is comparable for most tasks while the database size is substantially smaller. This confirms that the selected data retain the core predictive signal.

# 7 RELATED WORK

**Learning over a relational database.** Statistical relational learning [47, 96] is one of the first approaches to directly leverage the relational structure, factorized learning [66, 92, 102] avoids costly joins, automatic feature engineering [26, 62, 109, 129] synthesizes new features, while other approaches train arbitrary (often tree-based [22]) predictive models. The currently emerging approach is Relational Deep Learning (RDL) [29, 40, 114]. It models the relational database as a heterogeneous graph, and employs tabular feature encoders [58] trained jointly with Graph Neural Networks (GNNs) to solve a wide range of predictive tasks. The current research frontier further builds specialized architectures for RDL that fall under the same GNN paradigm [23, 55, 127] and adapts Graph Transformers to the relational setting [35]. Of course, the RDL approach inherits the opacity of "black-box" deep learning, creating the need for explainability. Since RDL models, irrespective of their implementation, make predictions at the database level, we consider explanations that refer to the database rather than the specific modeling components, such as the heterogeneous graph.

**ML explanations.** To address the explainability in RDL, we review the broader XAI literature. Explanations are categorized as either *local* (i.e., focusing on individual predictions) or *global* (i.e., explaining a model's overall behavior) [14, 53, 88, 101]. For example, local feature attribution methods [76, 97, 110], explain individual predictions by assigning importance scores to each feature associated with a specific instance, while global feature importance methods like PFI [15] and its variants [32, 42, 89], detect important features across all predictions by measuring the each feature's impact on model performance. A more formal distinction further classifies these methods into *abductive* and *contrastive* [13, 82]. Abductive explanations identify minimal subsets of the input *sufficient* to preserve the model's prediction, whereas contrastive explanations specify the minimal changes required to alter it. In our context, explanation views are global, as they refer to the database as a whole, and abductive, as they identify the database components sufficient to determine the model's behavior. Crucially, traditional XAI assumes a feature vector associated with each prediction instance. In RDL, there is no single "prediction tuple" containing all relevant features. Instead, predictions use the relational graph where tuples across multiple tables are connected, and features are derived through message-passing and aggregation. This makes these methods conceptually misaligned with RDL, as they ignore that features are computed by combining multiple connected tuples and that the relational structure itself carries predictive information.

**Subgraph-based GNN explanations.** *Instance-level* (i.e., local) explanations specific to Graph Neural Networks (GNNs) aim to answer questions of the type "Why does the GNN produce a particular output *for a given instance* (i.e., node in the graph)?". Most of these explanations are at the level of individual graph elements, i.e., the explanation is a subgraph of the input graph. Some instance-level explainers focus on neighboring nodes and their individual features [45, 68, 95, 113], others on important edges [75, 122], while others on connected subgraphs [70, 71, 87, 108, 126]. Few are tailored to hetero-GNNs [68, 87] by specializing subgraph explanations to meta-paths. Similarly, *global* GNN explainers, which provide insights for a GNN across all instances, use subgraphs as explanations. More specifically, some recover a collection of subgraphs that collectively explain many instances [24, 65, 77, 117, 124] aiming at explanation generality, while others extract instance-level explanation subgraphs which are then combined into global insights [7, 24, 78]. In contrast to our approach, most global explainers implicitly assume the predictive tasks are motif-based [9] and they are often evaluated against a ground truth. We refer the reader to a recent survey for more details on these techniques [125]. We depart from all these subgraph-based approaches in the following ways: (1) Our explanations are expressed in terms of the relational database, which is the input object, instead of the graph that is part of modeling. (2) The complexity of our explanations is measured on

**Table 2: Databases, tasks, model performance, and performance after training with only the data in our explanations.**

| Database, Task | D*i*T*j* | Perf. | Masked Perf. | Diff | % Removed |
|---|---|---|---|---|---|
| **Binary Classification (ROC-AUC)** | | | | | |
| rel-f1, driver-dnf | D1T1 | 75.91 | 73.27 | -2.64 | 51.19% |
| rel-f1, driver-top3 | D1T2 | 76.50 | 67.29 | -9.21 | 55.49% |
| rel-avito, user-clicks | D2T1 | 65.03 | 67.28 | 2.26 | 1.93% |
| rel-avito, user-visits | D2T2 | 64.45 | 63.64 | -0.81 | 8.34% |
| rel-stack, user-engage | D3T1 | 89.92 | 89.60 | -0.32 | 64.56% |
| rel-stack, user-badge | D3T2 | 88.33 | 88.11 | -0.23 | 84.05% |
| rel-hm, user-churn | D4T1 | 68.79 | 68.82 | 0.03 | 50.84% |
| rel-trial, study-outcome | D6T1 | 71.33 | 69.40 | -1.92 | 79.46% |
| rel-event, user-repeat | D5T1 | 75.45 | 78.60 | 3.16 | 94.81% |
| rel-event, user-ignore | D5T2 | 79.34 | 75.22 | -4.12 | 73.21% |
| **Regression (MAE)** | | | | | |
| rel-f1, driver-position | D1T3 | 3.2444 | 3.3487 | 0.1043 | 51.22% |
| rel-avito, ad-ctr | D2T3 | 0.0451 | 0.0457 | 0.0006 | 11.86% |
| rel-hm, item-sales | D4T2 | 0.0574 | 0.0566 | -0.0008 | 31.35% |
| rel-trial, study-adverse | D6T2 | 46.7445 | 45.4555 | -1.2890 | 85.23% |
| rel-trial, site-success | D6T3 | 0.3245 | 0.3747 | 0.0502 | 76.02% |
| rel-event, user-attendance | D5T3 | 0.2449 | 0.2381 | -0.0067 | 94.51% |

SQL views, which can succinctly refer to billions of data points. This implies that subgraph-based explanations can be expressed as SQL views, often much more succinctly than in graph-based representations. (3) Our SQL explanations can express database components that go beyond simple subgraphs. For example, a single-column projection does not correspond to a connected subgraph. It is also possible for explanations to refer to derived information, such as aggregates which do not explicitly exist in the graph at all.

**Other GNN explanations.** Beyond subgraphs, there are other types of explanations that are fundamentally different. First, some approaches, if adapted to RDL, would not correspond to any database elements [11, 59, 104–106, 113, 130], e.g., because they refer to GNN layers or construct surrogate models. Second, distillation [94] can map the GNN into a logical classifier. This is similar in spirit to our explanations; however, their base predicates refer to inner features of the model and can quickly become too complex. Finally, self-explainable GNNs [30] claim to provide simpler GNN models, thus with better interpretability properties. Recently this approach has been adapted to RDL, but it is limited to meta-path selection [39]. Our explanations are given post-hoc on an already trained model, instead of training a simpler model from scratch.

**Success metrics.** Most prior work focuses on sufficiency of explanations, i.e., finding a small subgraph that (by itself) retains the GNN prediction, quantified by a metric called *fidelity*[2] [125]. Variants of this metric either focus on retaining prediction confidence [68] or robustness under random, [132], infinitesimally small [2], and even adversarial [38] perturbations. In addition to sufficiency, some works assess explanation necessity [3, 74], i.e., whether the explanation subgraph is essential to maintain the prediction. Closest to soft determinacy is a fidelity variant called degree of sufficiency [8, 9] in that it has a probabilistic nature. Compared to all this prior work, our evaluation has the following key differences: (1) Since the framework is general, soft determinacy is parameterized by the explanation language **EL**. In contrast to subgraph modifications, the perturbations needed for soft determinacy are in the database space and must respect **EL** to be valid. (2) We minimize explanation complexity with respect to query instead of data size. (3) We also focus on the time to obtain an explanation, which has received limited attention in prior work, with few exceptions [24, 71, 126].

**Explanations in databases.** In the context of database research, explanations target various phenomena including why specific query results are present or not (why-not explanations) [20], outliers [86, 120], trends [100] or bias [25, 123] in query outputs, and performance anomalies [61, 128]. Diverse approaches have been employed: Provenance-based methods [48] attribute query results back to input tuples, with techniques ranging from deriving provenance expressions [4, 16, 51, 52, 107] to quantifying tuple contributions via Shapley values [34, 72, 73] or responsibility measures [44, 83, 84]. Intervention-based approaches perform targeted data perturbations (e.g., tuple deletion or addition) to assess causal impact on particular outcomes [99, 100, 120]. Summarization methods place a focus on conciseness, e.g., employing predicates [1, 37, 46, 67, 99, 100, 120] or taxonomies [49, 115] to succinctly describe significant parts of the database. Example-based techniques focus on minimal examples to enhance interpretability, e.g., highlighting errors via counterexamples [85]. These approaches are not mutually exclusive and can be combined, e.g., using summarization to refine complex provenance expressions [67, 111]. A complete taxonomy can be found in [50]. These efforts parallel XAI in aiming to explain algorithmic outputs via input data and/or computational logic. Differently, in our framework, explanations are themselves simple queries, explaining complex queries (deep models).

## 8 CONCLUSIONS AND FUTURE WORK

We presented a model-agnostic framework for abductive global explanations of ML models over relational databases, and studied its instantiation on hetero-GNNs, for which we proposed an efficient mask-learning approach. Our extensive evaluation on the RelBench benchmark demonstrated that our method discovers high-quality explanations with low runtime across diverse tasks.

Several directions for future work emerge from our framework that concern broadening the applicability and expressivity. One natural direction is to support additional explanation languages using aggregates, grouping, nesting, or more complex selection predicates. To achieve this, several interesting challenges arise, including defining appropriate database perturbations that respect the semantics of these constructs, systematically identifying good candidate predicates, and investigating efficient approaches for explanation discovery, since it is unclear whether mask-learning can be applied. Another direction is to develop efficient instantiations for different model classes, e.g., attention-based models [36, 56, 118], and different predictive tasks, e.g., recommendations. Beyond these, a fundamental question is to identify explanation languages that naturally align with a given use case; can we predict which explanation constructs will yield the most useful results for a particular combination of model architecture and database schema?

## ACKNOWLEDGMENTS

The work of Benny Kimelfeld was funded by the Israel Science Foundation (ISF) under grant 863/25.

[2]Fidelity is sometimes referred to as fidelity minus [3] or sufficiency [74]

## REFERENCES


[1] Firas Abuzaid, Peter Kraft, Sahaana Suri, Edward Gan, Eric Xu, Atul Shenoy, Asvin Ananthanarayan, John Sheu, Erik Meijer, Xi Wu, Jeff Naughton, Peter Bailis, and Matei Zaharia. 2018. DIFF: a relational interface for large-scale data explanation. *Proc. VLDB Endow.* 12, 4 (Dec. 2018), 419–432. https://doi.org/10.14778/3297753.3297761

[2] Chirag Agarwal, Marinka Zitnik, and Himabindu Lakkaraju. 2022. Probing GNN Explainers: A Rigorous Theoretical and Empirical Analysis of GNN Explanation Methods. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Vol. 151. PMLR, 8969–8996. https://proceedings.mlr.press/v151/agarwal22b.html

[3] Kenza Amara, Zhitao Ying, Zitao Zhang, Zhichao Han, Yang Zhao, Yinan Shan, Ulrik Brandes, Sebastian Schemm, and Ce Zhang. 2022. GraphFramEx: Towards Systematic Evaluation of Explainability Methods for Graph Neural Networks. In *Proceedings of the First Learning on Graphs Conference (Proceedings of Machine Learning Research)*, Vol. 198. PMLR, 44:1–44:23. https://proceedings.mlr.press/v198/amara22a.html

[4] Yael Amsterdamer, Daniel Deutch, and Val Tannen. 2011. Provenance for aggregate queries. In *Proceedings of the Thirtieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '11)*. Association for Computing Machinery, 153–164. https://doi.org/10.1145/1989284.1989302

[5] Plamen P. Angelov, Eduardo A. Soares, Richard Jiang, Nicholas I. Arnold, and Peter M. Atkinson. 2021. Explainable artificial intelligence: an analytical review. *WIREs Data Mining Knowl. Discov.* 11, 5 (2021). https://doi.org/10.1002/WIDM.1424

[6] Marcelo Arenas. 2024. A Data Management Approach to Explainable AI. In *PODS keynote*. 1–3. https://doi.org/10.1145/3635138.3654762

[7] Steve Azzolin, Antonio Longa, Pietro Barbiero, Pietro Lio, and Andrea Passerini. 2023. Global Explainability of GNNs via Logic Combination of Learned Concepts. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=OTbRTIY4YS

[8] Steve Azzolin, Antonio Longa, Stefano Teso, and Andrea Passerini. 2025. Reconsidering Faithfulness in Regular, Self-Explainable and Domain Invariant GNNs. In *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=kiOxNsrpQy

[9] Steve Azzolin, Sagar Malhotra, Andrea Passerini, and Stefano Teso. 2025. Beyond Topological Self-Explainable GNNs: A Formal Explainability Perspective. In *Forty-second International Conference on Machine Learning*. https://openreview.net/forum?id=mkqcUWBykZ

[10] Garima Bakshi, Rati Shukla, Vikash Yadav, Aman Dahiya, Rohit Anand, Nidhi Sindhwani, Harinder Singh, et al. 2021. An optimized approach for feature extraction in multi-relational statistical learning. *Journal of Scientific & Industrial Research* 80, 6 (2021), 537–542. https://doi.org/10.56042/jsir.v80i6.43632

[11] Federico Baldassarre and Hossein Azizpour. 2019. Explainability Techniques for Graph Convolutional Networks. In *International Conference on Machine Learning (ICML) Workshops, 2019 Workshop on Learning and Reasoning with Graph-Structured Representations*. https://arxiv.org/abs/1905.13686

[12] Pablo Barcelo, Mikhail Galkin, Christopher Morris, and Miguel Romero Orth. 2022. Weisfeiler and Leman Go Relational. In *Proceedings of the First Learning on Graphs Conference (Proceedings of Machine Learning Research)*, Vol. 198. PMLR, 46:1–46:26. https://proceedings.mlr.press/v198/barcelo22a.html

[13] Pablo Barceló, Alexander Kozachinskiy, Miguel Romero, Bernardo Subercaseaux, and José Verschae. 2025. Explaining k-Nearest Neighbors: Abductive and Counterfactual Explanations. *Proc. ACM Manag. Data* 3, 2, Article 97 (June 2025), 26 pages. https://doi.org/10.1145/3725234

[14] Shahaf Bassan, Guy Amir, and Guy Katz. 2024. Local vs. global interpretability: a computational complexity perspective. In *Proceedings of the 41st International Conference on Machine Learning (ICML'24)*. JMLR.org, Article 126, 35 pages. https://openreview.net/forum?id=324zEJCo3a

[15] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32. https://doi.org/10.1023/A:1010933404324

[16] Peter Buneman, Sanjeev Khanna, and Wang Chiew Tan. 2001. Why and Where: A Characterization of Data Provenance. In *Proceedings of the 8th International Conference on Database Theory (ICDT '01)*. Springer-Verlag, 316–330. https://doi.org/10.1007/3-540-44503-X_20

[17] Riccardo Cappuzzo, Paolo Papotti, and Saravanan Thirumuruganathan. 2020. Creating Embeddings of Heterogeneous Relational Datasets for Data Integration Tasks. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD '20)*. Association for Computing Machinery, 1335–1349. https://doi.org/10.1145/3318464.3389742

[18] Riccardo Cappuzzo, Saravanan Thirumuruganathan, and Paolo Papotti. 2024. Relational data imputation with graph neural networks. In *EDBT/ICDT 2024, 27th International Conference on Extending Database Technology, 25-28 March 2024, Paestum, Italy*. 221–233. https://doi.org/10.48786/EDBT.2024.20

[19] Stephen Casper, Tilman Rauker, Anson Ho, and Dylan Hadfield-Menell. 2023. SoK: Toward Transparent AI: A Survey on Interpreting the Inner Structures of Deep Neural Networks. In *First IEEE Conference on Secure and Trustworthy Machine Learning*. https://openreview.net/forum?id=8C5zt-0Utdn

[20] Adriane Chapman and H. V. Jagadish. 2009. Why not?. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD '09)*. Association for Computing Machinery, 523–534. https://doi.org/10.1145/1559845.1559901

[21] Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. 2018. Learning to Explain: An Information-Theoretic Perspective on Model Interpretation. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Vol. 80. PMLR, 883–892. https://proceedings.mlr.press/v80/chen18j.html

[22] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. Association for Computing Machinery, 785–794. https://doi.org/10.1145/2939672.2939785

[23] Tianlang Chen, Charilaos Kanatsoulis, and Jure Leskovec. 2025. RelGNN: Composite Message Passing for Relational Deep Learning. In *Forty-second International Conference on Machine Learning*. https://openreview.net/forum?id=XXh3zmw2Uy

[24] Tingyang Chen, Dazhuo Qiu, Yinghui Wu, Arijit Khan, Xiangyu Ke, and Yunjun Gao. 2024. View-based Explanations for Graph Neural Networks. *Proc. ACM Manag. Data* 2, 1, Article 40 (March 2024), 27 pages. https://doi.org/10.1145/3639295

[25] Zixuan Chen, Panagiotis Manolios, and Mirek Riedewald. 2023. Why Not Yet: Fixing a Top-k Ranking that is Not Fair to Individuals. *Proc. VLDB Endow.* 16, 9 (May 2023), 2377–2390. https://doi.org/10.14778/3598581.3598606

[26] Nadiia Chepurko, Ryan Marcus, Emanuel Zgraggen, Raul Castro Fernandez, Tim Kraska, and David Karger. 2020. ARDA: automatic relational data augmentation for machine learning. *Proc. VLDB Endow.* 13, 9 (May 2020), 1373–1387. https://doi.org/10.14778/3397230.3397235

[27] Clinical Trials Transformation Initiative (CTTI). [n.d.]. Aggregate Analysis of ClinicalTrials.gov (AACT) Database. https://aact.ctti-clinicaltrials.org/ Accessed: 2025.

[28] Benoît Colson, Patrice Marcotte, and Gilles Savard. 2007. An overview of bilevel optimization. *Annals of operations research* 153, 1 (2007), 235–256. https://doi.org/10.1007/s10479-007-0176-2

[29] Milan Cvitkovic. 2019. Supervised learning on relational databases with graph neural networks. In *International Conference on Machine Learning (ICML) Workshops, 2019 Workshop on Representation Learning on Graphs and Manifolds*. https://arxiv.org/abs/2002.02046

[30] Enyan Dai and Suhang Wang. 2021. Towards Self-Explainable Graph Neural Network. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21)*. Association for Computing Machinery, 302–311. https://doi.org/10.1145/3459637.3482306

[31] Adnan Darwiche. 2023. Logic for Explainable AI. In *Proceedings of the Thirty eighth Annual IEEE Symposium on Logic in Computer Science (LICS 2023)*. IEEE Computer Society Press, 1–11. https://doi.org/10.1109/LICS56636.2023.10175757

[32] Dries Debeer and Carolin Strobl. 2020. Conditional permutation importance revisited. *BMC bioinformatics* 21, 1 (2020), 307. https://doi.org/10.1186/s12859-020-03622-2

[33] Stephan Dempe. 2002. *Foundations of bilevel programming*. Springer. https://doi.org/10.1007/b101970

[34] Daniel Deutch, Nave Frost, Benny Kimelfeld, and Mikaël Monet. 2022. Computing the Shapley Value of Facts in Query Answering. In *Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22)*. Association for Computing Machinery, 1570–1583. https://doi.org/10.1145/3514221.3517912

[35] Vijay Prakash Dwivedi, Sri Jaladi, Yangyi Shen, Federico Lopez, Charilaos I. Kanatsoulis, Rishi Puri, Matthias Fey, and Jure Leskovec. 2026. Relational Graph Transformer. In *The Fourteenth International Conference on Learning Representations*. https://openreview.net/forum?id=2d3j6bt21A

[36] Vijay Prakash Dwivedi, Charilaos Kanatsoulis, Shenyang Huang, and Jure Leskovec. 2025. Relational Deep Learning: Challenges, Foundations and Next-Generation Architectures. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25)*. Association for Computing Machinery, 5999–6009. https://doi.org/10.1145/3711896.3736558

[37] Kareem El Gebaly, Parag Agrawal, Lukasz Golab, Flip Korn, and Divesh Srivastava. 2014. Interpretable and informative explanations of outcomes. *Proc. VLDB Endow.* 8, 1 (Sept. 2014), 61–72. https://doi.org/10.14778/2735461.2735467

[38] Junfeng Fang, Wei Liu, Yuan Gao, Zemin Liu, An Zhang, Xiang Wang, and Xiangnan He. 2023. Evaluating Post-hoc Explanations for Graph Neural Networks via Robustness Analysis. In *Advances in Neural Information Processing Systems*, Vol. 36. Curran Associates, Inc., 72446–72463. https://proceedings.neurips.cc/paper_files/paper/2023/file/e55c2f3fdde519014c879aa3554414c0-Paper-Conference.pdf

[39] Francesco Ferrini, Antonio Longa, Andrea Passerini, and Manfred Jaeger. 2025. A Self-Explainable Heterogeneous GNN for Relational Deep Learning. *Transactions on Machine Learning Research* (2025). https://openreview.net/forum?id=8Q4qxe9a9Z

[40] Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. 2024. Position: Relational Deep Learning - Graph Representation Learning on Relational Databases. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Vol. 235. PMLR, 13592–13607. https://proceedings.mlr.press/v235/fey24a.html

[41] Matthias Fey, Jinu Sunil, Akihiro Nitta, Rishi Puri, Manan Shah, Blaž Stojanovič, Ramona Bendias, Alexandria Barghi, Vid Kocijan, Zecheng Zhang, Xinwei He, Jan Eric Lenssen, and Jure Leskovec. 2025. PyG 2.0: Scalable Learning on Real World Graphs. In *Temporal Graph Learning Workshop @ KDD 2025*. https://openreview.net/forum?id=DHHLkQvWqs

[42] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. 2019. All Models are Wrong, but Many are Useful: Learning a Variable's Importance by Studying an Entire Class of Prediction Models Simultaneously. *Journal of Machine Learning Research* 20, 177 (2019), 1–81. http://jmlr.org/papers/v20/18-760.html

[43] Ruth C. Fong and Andrea Vedaldi. 2017. Interpretable Explanations of Black Boxes by Meaningful Perturbation. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 3449–3457. https://doi.org/10.1109/ICCV.2017.371

[44] Cibele Freire, Wolfgang Gatterbauer, Neil Immerman, and Alexandra Meliou. 2015. The complexity of resilience and responsibility for self-join-free conjunctive queries. *Proc. VLDB Endow.* 9, 3 (Nov. 2015), 180–191. https://doi.org/10.14778/2850583.2850592

[45] Thorben Funke, Megha Khosla, and Avishek Anand. 2021. Hard Masking for Explaining Graph Neural Networks. https://openreview.net/forum?id=uDN8pRAdsoC

[46] Karim E. Gebaly, Ge Feng, Lukasz Golab, Flip Korn, and Divesh Srivastava. 2018. Explanation Tables. *IEEE Data Engineering Bulletin* 41, 3 (2018), 43–51. http://sites.computer.org/debull/A18sept/p43.pdf

[47] Lise Getoor and Ben Taskar. 2007. *Introduction to Statistical Relational Learning*. MIT press. https://doi.org/10.7551/mitpress/7432.001.0001

[48] Boris Glavic. 2021. Data Provenance - Origins, Applications, Algorithms, and Models. *Foundations and Trends® in Databases* 9, 3-4 (2021), 209–441. https://doi.org/10.1561/1900000068

[49] Boris Glavic, Sven Köhler, Sean Riddle, and Bertram Ludäscher. 2015. Towards Constraint-based Explanations for Answers and Non-Answers. In *7th USENIX Workshop on the Theory and Practice of Provenance (TaPP 15)*. USENIX Association. https://www.usenix.org/conference/tapp15/workshop-program/presentation/glavic

[50] Boris Glavic, Alexandra Meliou, and Sudeepa Roy. 2021. Trends in Explanations: Understanding and Debugging Data-driven Systems. *Found. Trends Databases* 11, 3 (Aug. 2021), 226–318. https://doi.org/10.1561/1900000074

[51] Boris Glavic, Renée J. Miller, and Gustavo Alonso. 2013. *Using SQL for Efficient Generation and Querying of Provenance Information*. Springer Berlin Heidelberg, 291–320. https://doi.org/10.1007/978-3-642-41660-6_16

[52] Todd J. Green and Val Tannen. 2017. The Semiring Framework for Database Provenance. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS '17)*. Association for Computing Machinery, 93–99. https://doi.org/10.1145/3034786.3056125

[53] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A Survey of Methods for Explaining Black Box Models. *ACM Comput. Surv.* 51, 5, Article 93 (Aug. 2018), 42 pages. https://doi.org/10.1145/3236009

[54] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. 2015. Statistical learning with sparsity. *Monographs on statistics and applied probability* 143, 143 (2015), 8. https://dl.acm.org/doi/10.5555/2834535

[55] Benjamin Hilprecht, Kristian Kersting, and Carsten Binnig. 2023. SPARE: A Single-Pass Neural Model for Relational Databases. *arXiv preprint arXiv:2310.13581* (2023). https://arxiv.org/abs/2310.13581

[56] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. 2025. Accurate predictions on small data with a tabular foundation model. *Nature* 637 (2025), 319–326. https://doi.org/10.1038/s41586-024-08328-6

[57] Giles Hooker, Lucas Mentch, and Siyu Zhou. 2021. Unrestricted permutation forces extrapolation: variable importance requires at least one more model, or there is no free variable importance. *Statistics and Computing* 31, 6 (Nov. 2021), 16. https://doi.org/10.1007/s11222-021-10057-z

[58] Weihua Hu, Yiwen Yuan, Zecheng Zhang, Akihiro Nitta, Kaidi Cao, Vid Kocijan, Jinu Sunil, Jure Leskovec, and Matthias Fey. 2024. PyTorch Frame: A Modular Framework for Multi-Modal Tabular Learning. In *NeurIPS 2024 Third Table Representation Learning Workshop*. https://openreview.net/forum?id=2ZHKA9xo8V

[59] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, and Yi Chang. 2023. GraphLIME: Local Interpretable Model Explanations for Graph Neural Networks. *IEEE Trans. on Knowl. and Data Eng.* 35, 7 (July 2023), 6968–6972. https://doi.org/10.1109/TKDE.2022.3187455

[60] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rkE3y85ee

[61] Prajakta Kalmegh, Shivnath Babu, and Sudeepa Roy. 2018. iQCAR: Inter-Query Contention Analyzer. In *Proceedings of the ACM Symposium on Cloud Computing (SoCC '18)*. Association for Computing Machinery, 532. https://doi.org/10.1145/3267809.3275473

[62] James Max Kanter and Kalyan Veeramachaneni. 2015. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE international conference on data science and advanced analytics (DSAA)*. IEEE, 1–10. https://doi.org/10.1109/DSAA.2015.7344858

[63] Benny Kimelfeld and Christopher Ré. 2018. A Relational Framework for Classifier Engineering. *SIGMOD Record* 47, 1 (2018), 6–13. https://doi.org/10.1145/3277006.3277009

[64] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *The Third International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.1412.6980

[65] Mert Kosan, Zexi Huang, Sourav Medya, Sayan Ranu, and Ambuj Singh. 2025. GCFExplainer: Global Counterfactual Explainer for Graph Neural Networks. *ACM Trans. Intell. Syst. Technol.* 16, 5, Article 99 (Aug. 2025), 23 pages. https://doi.org/10.1145/3698108

[66] Arun Kumar, Jeffrey Naughton, Jignesh M. Patel, and Xiaojin Zhu. 2016. To Join or Not to Join? Thinking Twice about Joins before Feature Selection. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*. Association for Computing Machinery, 19–34. https://doi.org/10.1145/2882903.2882952

[67] Seokki Lee, Bertram Ludäscher, and Boris Glavic. 2020. Approximate summaries for why and why-not provenance. *Proc. VLDB Endow.* 13, 6 (Feb. 2020), 912–924. https://doi.org/10.14778/3380750.3380760

[68] Tong Li, Jiale Deng, Yanyan Shen, Luyu Qiu, Yongxiang Huang, and Caleb Chen Cao. 2023. Towards fine-grained explainability for heterogeneous graph neural network. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence (AAAI'23/IAAI'23/EAAI'23)*. AAAI Press, Article 971, 8 pages. https://doi.org/10.1609/aaai.v37i7.26040

[69] Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A Logic-Driven Framework for Consistency of Neural Models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 3924–3935. https://doi.org/10.18653/v1/D19-1405

[70] Wenqian Li, Yinchuan Li, Zhigang Li, Jianye HAO, and Yan Pang. 2023. DAG Matters! GFlowNets Enhanced Explainer for Graph Neural Networks. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=jgmuRzM-sb6

[71] Wanyu Lin, Hao Lan, and Baochun Li. 2021. Generative Causal Explanations for Graph Neural Networks. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Vol. 139. PMLR, 6666–6679. https://proceedings.mlr.press/v139/lin21d.html

[72] Ester Livshits, Leopoldo Bertossi, Benny Kimelfeld, and Moshe Sebag. 2020. The Shapley Value of Tuples in Query Answering. In *23rd International Conference on Database Theory (ICDT 2020) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 155. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 20:1–20:19. https://doi.org/10.4230/LIPIcs.ICDT.2020.20

[73] Ester Livshits, Leopoldo Bertossi, Benny Kimelfeld, and Moshe Sebag. 2021. Query Games in Databases. *SIGMOD Rec.* 50, 1 (June 2021), 78–85. https://doi.org/10.1145/3471485.3471504

[74] Antonio Longa, Steve Azzolin, Gabriele Santin, Giulia Cencetti, Pietro Lio, Bruno Lepri, and Andrea Passerini. 2025. Explaining the Explainers in Graph Neural Networks: a Comparative Study. *ACM Comput. Surv.* 57, 5, Article 120 (Jan. 2025), 37 pages. https://doi.org/10.1145/3696444

[75] Ana Lucic, Maartje A. Ter Hoeve, Gabriele Tolomei, Maarten De Rijke, and Fabrizio Silvestri. 2022. CF-GNNExplainer: Counterfactual Explanations for Graph Neural Networks. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Vol. 151. PMLR, 4499–4511. https://proceedings.mlr.press/v151/lucic22a.html

[76] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

[77] Ge Lv and Lei Chen. 2023. On Data-Aware Global Explainability of Graph Neural Networks. *Proc. VLDB Endow.* 16, 11 (July 2023), 3447–3460. https://doi.org/10.14778/3611479.3611538

[78] Ge Lv, Lei Chen, and Caleb Chen Cao. 2022. On Glocal Explainability of Graph Neural Networks. In *Database Systems for Advanced Applications: 27th International Conference, DASFAA 2022, Virtual Event, April 11–14, 2022, Proceedings, Part I*. Springer-Verlag, 648–664. https://doi.org/10.1007/978-3-031-00123-9_52

[79] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *International Conference on Learning Representations*. https://openreview.net/forum?id=S1jE5L5gl

[80] Chris J. Maddison, Daniel Tarlow, and Tom Minka. 2014. A* Sampling. In *Advances in Neural Information Processing Systems*, Vol. 27. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2014/file/937debc749f041eb5700df7211ac795c-Paper.pdf

[81] Joao Marques-Silva. 2023. *Logic-Based Explainability in Machine Learning*. Springer Nature Switzerland, 24–104. https://doi.org/10.1007/978-3-031-31414-8_2

[82] Joao Marques-Silva and Alexey Ignatiev. 2022. Delivering trustworthy AI through formal XAI. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 12342–12350. https://doi.org/10.1609/aaai.v36i11.21499

[83] Alexandra Meliou, Wolfgang Gatterbauer, Katherine F. Moore, and Dan Suciu. 2010. The complexity of causality and responsibility for query answers and non-answers. *Proc. VLDB Endow.* 4, 1 (Oct. 2010), 34–45. https://doi.org/10.14778/1880172.1880176

[84] Alexandra Meliou, Wolfgang Gatterbauer, and Dan Suciu. 2011. Reverse data management. *Proc. VLDB Endow.* 4, 12 (Aug. 2011), 1490–1493. https://doi.org/10.14778/3402755.3402803

[85] Zhengjie Miao, Sudeepa Roy, and Jun Yang. 2019. Explaining Wrong Queries Using Small Examples. In *Proceedings of the 2019 International Conference on Management of Data (SIGMOD '19)*. Association for Computing Machinery, 503–520. https://doi.org/10.1145/3299869.3319866

[86] Zhengjie Miao, Qitian Zeng, Chenjie Li, Boris Glavic, Oliver Kennedy, and Sudeepa Roy. 2019. CAPE: explaining outliers by counterbalancing. *Proc. VLDB Endow.* 12, 12 (Aug. 2019), 1806–1809. https://doi.org/10.14778/3352063.3352071

[87] Grzegorz P. Mika, Amel Bouzeghoub, Katarzyna Węgrzyn-Wolska, and Yessin M. Neggaz. 2023. HGExplainer: Explainable Heterogeneous Graph Neural Network. In *2023 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. 221–229. https://doi.org/10.1109/WI-IAT59888.2023.00035

[88] Christoph Molnar. 2025. *Interpretable Machine Learning* (3rd ed.). https://christophm.github.io/interpretable-ml-book Accessed: 2025.

[89] Christoph Molnar, Gunnar König, Bernd Bischl, and Giuseppe Casalicchio. 2023. Model-agnostic feature importance and effects with dependent features: a conditional subgroup approach. *Data Min. Knowl. Discov.* 38, 5 (Jan. 2023), 2903–2941. https://doi.org/10.1007/s10618-022-00901-9

[90] Luis Müller, Mikhail Galkin, Christopher Morris, and Ladislav Rampášek. 2024. Attending to Graph Transformers. *Transactions on Machine Learning Research* 2024 (2024). https://openreview.net/forum?id=HhbqHBBrfZ

[91] Alan Nash, Luc Segoufin, and Victor Vianu. 2010. Views and queries: Determinacy and rewriting. *ACM Trans. Database Syst.* 35, 3, Article 21 (July 2010), 41 pages. https://doi.org/10.1145/1806907.1806913

[92] Dan Olteanu. 2020. The relational data borg is learning. *Proc. VLDB Endow.* 13, 12 (Aug. 2020), 3502–3515. https://doi.org/10.14778/3415478.3415572

[93] Claudia Perlich and Foster J. Provost. 2003. Aggregation-based feature invention and relational concept classes. In *KDD*. 167–176. https://doi.org/10.1145/956750.956772

[94] Alexander Pluska, Pascal Welke, Thomas Gärtner, and Sagar Malhotra. 2024. Logical distillation of graph neural networks. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning (KR '24)*. Article 86, 11 pages. https://doi.org/10.24963/kr.2024/86

[95] Phillip E. Pope, Soheil Kolouri, Mohammad Rostami, Charles E. Martin, and Heiko Hoffmann. 2019. Explainability Methods for Graph Convolutional Neural Networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10764–10773. https://doi.org/10.1109/CVPR.2019.01103

[96] Luc Raedt, Kristian Kersting, Sriraam Natarajan, and David Poole. 2016. Statistical Relational Artificial Intelligence: Logic, Probability, and Computation. https://link.springer.com/book/10.1007/978-3-031-01574-8

[97] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. Association for Computing Machinery, 1135–1144. https://doi.org/10.1145/2939672.2939778

[98] Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan E. Lenssen, Yiwen Yuan, Zecheng Zhang, Xinwei He, and Jure Leskovec. 2024. RelBench: A Benchmark for Deep Learning on Relational Databases. In *Advances in Neural Information Processing Systems*, Vol. 37. Curran Associates, Inc., 21330–21341. https://doi.org/10.52202/079017-0672

[99] Sudeepa Roy, Laurel Orr, and Dan Suciu. 2015. Explaining query answers with explanation-ready databases. *Proc. VLDB Endow.* 9, 4 (Dec. 2015), 348–359. https://doi.org/10.14778/2856318.2856329

[100] Sudeepa Roy and Dan Suciu. 2014. A formal approach to finding explanations for database queries. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (SIGMOD '14)*. Association for Computing Machinery, 1579–1590. https://doi.org/10.1145/2588555.2588578

[101] Wojciech Samek and Klaus-Robert Müller. 2019. Towards explainable artificial intelligence. In *Explainable AI: interpreting, explaining and visualizing deep learning*. Springer, 5–22. https://dl.acm.org/doi/10.5555/3365202

[102] Maximilian Schleich, Dan Olteanu, and Radu Ciucanu. 2016. Learning Linear Regression Models over Factorized Joins. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*. Association for Computing Machinery, 3–18. https://doi.org/10.1145/2882903.2882939

[103] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*. Springer, 593–607. https://doi.org/10.1007/978-3-319-93417-4_38

[104] Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. 2021. Interpreting Graph Neural Networks for {NLP} With Differentiable Edge Masking. In *International Conference on Learning Representations*. https://openreview.net/forum?id=WznmQa42ZAx

[105] Thomas Schnake, Oliver Eberle, Jonas Lederer, Shinichi Nakajima, Kristof T. Schütt, Klaus-Robert Müller, and Grégoire Montavon. 2022. Higher-Order Explanations of Graph Neural Networks via Relevant Walks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 11 (2022), 7581–7596. https://doi.org/10.1109/TPAMI.2021.3115452

[106] Robert Schwarzenberg, Marc Hübner, David Harbecke, Christoph Alt, and Leonhard Hennig. 2019. Layerwise Relevance Visualization in Convolutional Text Graph Classifiers. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*. Association for Computational Linguistics, 58–62. https://doi.org/10.18653/v1/D19-5308

[107] Pierre Senellart. 2018. Provenance and Probabilities in Relational Databases. *SIGMOD Rec.* 46, 4 (Feb. 2018), 5–15. https://doi.org/10.1145/3186549.3186551

[108] Caihua Shan, Yifei Shen, Yao Zhang, Xiang Li, and Dongsheng Li. 2021. Reinforcement Learning Enhanced Explainer for Graph Neural Networks. In *Advances in Neural Information Processing Systems*, Vol. 34. Curran Associates, Inc., 22523–22533. https://proceedings.neurips.cc/paper_files/paper/2021/file/be26abe76fb5c8a4921cf9d3e865b454-Paper.pdf

[109] Boris Stanoev, Goran Mitrov, Andrea Kulakov, Georgina Mirceva, Petre Lameski, and Eftim Zdravevski. 2024. Automating Feature Extraction from Entity-Relation Models: Experimental Evaluation of Machine Learning Methods for Relational Learning. *Big Data and Cognitive Computing* 8, 4 (2024). https://doi.org/10.3390/bdcc8040039

[110] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Vol. 70. PMLR, 3319–3328. https://proceedings.mlr.press/v70/sundararajan17a.html

[111] Balder ten Cate, Cristina Civili, Evgeny Sherkhonov, and Wang-Chiew Tan. 2015. High-Level Why-Not Explanations using Ontologies. In *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS '15)*. Association for Computing Machinery, 31–43. https://doi.org/10.1145/2745754.2745765

[112] Jan Tönshoff, Neta Friedman, Martin Grohe, and Benny Kimelfeld. 2023. Stable Tuple Embeddings for Dynamic Databases. In *ICDE*. IEEE, 1286–1299. https://doi.org/10.1109/ICDE55515.2023.00103

[113] Minh Vu and My T. Thai. 2020. PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 12225–12235. https://proceedings.neurips.cc/paper_files/paper/2020/file/8fb134f258b1f7865a6ab2d935a897c9-Paper.pdf

[114] Minjie Wang, Quan Gan, David Wipf, Zhenkun Cai, Ning Li, Jianheng Tang, Yanlin Zhang, Zizhao Zhang, Zunyao Mao, Yakun Song, Yanbo Wang, Jiahang Li, Han Zhang, Guang Yang, Xiao Qin, Chuan Lei, Muhan Zhang, Weinan Zhang, Christos Faloutsos, and Zheng Zhang. 2024. 4DBInfer: A 4D Benchmarking Toolbox for Graph-Centric Predictive Modeling on RDBs. In *Advances in Neural Information Processing Systems*, Vol. 37. Curran Associates, Inc., 27236–27273. https://doi.org/10.52202/079017-0856

[115] Xiaolan Wang, Xin Luna Dong, and Alexandra Meliou. 2015. Data X-Ray: A Diagnostic Tool for Data Errors. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD '15)*. Association for Computing Machinery, 1231–1245. https://doi.org/10.1145/2723372.2750549

[116] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous Graph Attention Network. In *The World Wide Web Conference (WWW '19)*. Association for Computing Machinery, 2022–2032. https://doi.org/10.1145/3308558.3313562

[117] Xiaoqi Wang and Han Wei Shen. 2023. GNNInterpreter: A Probabilistic Generative Model-Level Explanation for Graph Neural Networks. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=rqq6Dh8t4d

[118] Yanbo Wang, Xiyuan Wang, Quan Gan, Minjie Wang, Qibin Yang, David Wipf, and Muhan Zhang. 2025. Griffin: Towards a Graph-Centric Relational Database Foundation Model. In *Forty-second International Conference on Machine Learning*. https://openreview.net/forum?id=TxeCxVb3cL

[119] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.* 8, 3–4 (May 1992), 229–256. https://doi.org/10.1007/BF00992696

[120] Eugene Wu and Samuel Madden. 2013. Scorpion: explaining away outliers in aggregate queries. *Proc. VLDB Endow.* 6, 8 (June 2013), 553–564. https:

//doi.org/10.14778/2536354.2536356

[121] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2020), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386

[122] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNNExplainer: Generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2019/file/d80b7040b773199015de6d3b4293c8ff-Paper.pdf

[123] Brit Youngmann, Michael Cafarella, Yuval Moskovitch, and Babak Salimi. 2023. On Explaining Confounding Bias. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. 1846–1859. https://doi.org/10.1109/ICDE55515.2023.00144

[124] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. 2020. XGNN: Towards Model-Level Explanations of Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*. Association for Computing Machinery, 430–438. https://doi.org/10.1145/3394486.3403085

[125] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. 2023. Explainability in Graph Neural Networks: A Taxonomic Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 5 (May 2023), 5782–5799. https://doi.org/10.1109/TPAMI.2022.3204236

[126] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. 2021. On Explainability of Graph Neural Networks via Subgraph Explorations. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Vol. 139. PMLR, 12241–12252. https://proceedings.mlr.press/v139/yuan21c.html

[127] Yiwen Yuan, Zecheng Zhang, Xinwei He, Akihiro Nitta, Weihua Hu, Manan Shah, Blaž Stojanovič, Shenyang Huang, Jan Eric Lenssen, Jure Leskovec, and Matthias Fey. 2025. ContextGNN: Beyond Two-Tower Recommendation Systems. In *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=nzOD1we8Z4

[128] Haopeng Zhang, Yanlei Diao, and Alexandra Meliou. 2017. Exstream: Explaining anomalies in event stream monitoring. In *Proceedings of the 20th international conference on extending database technology (EDBT)*. 156–167. https://doi.org/10.5441/002/edbt.2017.15

[129] Han Zhang, Quan Gan, David Wipf, and Weinan Zhang. 2023. Gfs: Graph-based feature synthesis for prediction over relational databases. *arXiv preprint arXiv:2312.02037* (2023). https://arxiv.org/abs/2312.02037

[130] Yue Zhang, David Defazio, and Arti Ramesh. 2021. RelEx: A Model-Agnostic Relational Model Explainer. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society (AIES '21)*. Association for Computing Machinery, 1042–1049. https://doi.org/10.1145/3461702.3462562

[131] Jianan Zhao, Xiao Wang, Chuan Shi, Binbin Hu, Guojie Song, and Yanfang Ye. 2021. Heterogeneous Graph Structure Learning for Graph Neural Networks. In *AAAI*. AAAI Press, 4697–4705. https://doi.org/10.1609/aaai.v35i5.16600

[132] Xu Zheng, Farhad Shirani, Tianchun Wang, Wei Cheng, Zhuomin Chen, Haifeng Chen, Hua Wei, and Dongsheng Luo. 2024. Towards Robust Fidelity for Evaluating Explainability of Graph Neural Networks. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=up6hr4hIQH

[133] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. 2017. Visualizing Deep Neural Network Decisions: Prediction Difference Analysis. In *International Conference on Learning Representations*. https://openreview.net/forum?id=BJ5UeU9xx