# Representing Multi-Relational Data via Statistical Relational Learning and Graph Convolutional Networks

Devendra Singh Dhami*
Eindhoven University of Technology
Netherlands
d.s.dhami@tue.nl

Saurabh Mathur*
Technical University of Darmstadt
Germany
saurabh.mathur@tu-darmstadt.de

Siwen Yan
ByteDance
USA
siwen.yan@utdallas.edu

Sriraam Natarajan
The University of Texas at Dallas
USA
sriraam.natarajan@utdallas.edu

## Abstract

Learning and reasoning about a graph's entities and relations requires informative representations. Typical graph embedding methods fail to fully capture the symmetries contained in the graph, making them sample inefficient. Moreover, the embeddings learned by these methods are opaque, limiting their applicability in high-stakes domains like identifying drug interactions. We consider the problem of learning rich yet interpretable graph representations from smaller graphs by exploiting their rich multi-relational structure. Specifically, we propose a solution based on statistical relational rule learning. We construct edge representations by learning relational logic rules to predict edges, applying the rules to the edges, and counting the number of ways each rule can be satisfied. We use these edge-based graph representations to construct graph convolutional networks (GCNs) by replacing the input graph's adjacency matrix with an edge distance matrix. Our comprehensive empirical evaluation demonstrates the superiority of our method over multiple approaches, including graph embedding techniques, variations of GCNs, and rule learning techniques.

## CCS Concepts

• **Computing methodologies** → **Statistical relational learning**; **Neural networks**.

## Keywords

Graph Convolutional Networks, First Order Logic, Statistical Relational Learning

**ACM Reference Format:**
Devendra Singh Dhami, Saurabh Mathur, Siwen Yan, and Sriraam Natarajan. 2025. Representing Multi-Relational Data via Statistical Relational Learning and Graph Convolutional Networks. In *Proceedings of International Conference on Data Science (CODS 2025).* ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

*Equal contribution


*CODS 2025, Pune, India*

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

## 1 Introduction

Graphs are ubiquitous data structures for representing relationships and interactions in a variety of domains, including social networks [29], chemical interactions [11], biological systems [43], knowledge bases [3], and recommendation systems [28]. They consist of nodes and edges; this structure provides a natural way to model complex, structured data and capture intricate dependencies often lost in traditional tabular representations. However, the raw structure of a graph is insufficient for learning tasks such as node classification and edge/link prediction since the graph structure lacks explicit feature representations encoding the semantic and structural properties of nodes and edges. This has spurred significant interest in representation learning on graphs, which aims to derive meaningful, low-dimensional embeddings that preserve the graph's topological and relational information.

One such approach to graph representation learning is graph convolution networks (GCNs, [21]). They extend convolutional neural networks (CNNs, [9]) from images to graphs. Despite their promise, GCNs face several challenges, particularly in high-stakes domains, where the diversity of relationship types and the complexity of underlying patterns complicate the construction of effective node representations. As a result, practitioners often resort to graph embedding learning methods [6] to construct the input features for GCNs. These methods learn embedding vectors for each node in the graph by extending word embedding methods from text to graphs. These methods suffer from three key limitations: (1) learning requires large amounts of data, (2) the learned node embeddings are opaque, and (3) they struggle to generalize to new entities.

We address these limitations of graph embedding methods through Statistical Relational Learning (SRL, [24]). SRL methods learn probabilistic logic rules, combining first-order logic's ability to faithfully capture rich domain structure with probabilistic models' ability to deal with uncertainty. While these models are interpretable, their scalability is limited by the difficulty of inference. We aim to combine the scalability of deep representation learning with the rich yet interpretable representations of SRL. To this effect, we propose *relat**I**onal de**N**sity dis**T**ance bas**E**d **GRA**ph convolu**T**ional*

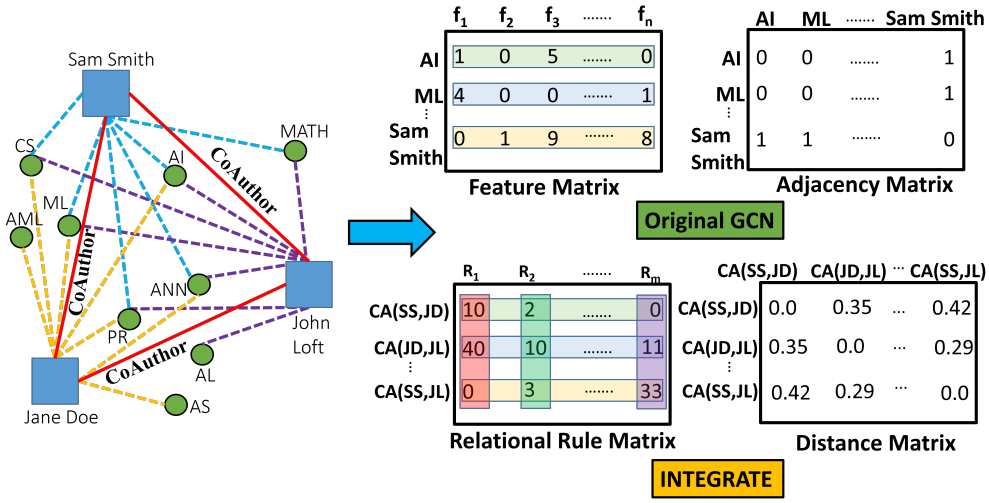


**Figure 1: Example: An incomplete graph representing co-authorship (CA) between researchers (left). The graph might be completed by learning a graph convolutional network (GCN) to predict missing relations; this requires a feature matrix and an adjacency matrix representing edges between nodes. Existing GCN approaches (top-right) use opaque node embeddings to construct the feature matrix. INTEGRATE (bottom-right) constructs interpretable edge features; each feature corresponding to a first-order logic rule. We also replace the node-adjacency matrix with an edge distance matrix, constructed using our edge features.**

*n**E**tworks* (INTEGRATE)[1], a sample-efficient method for learning interpretable yet rich features from graphs. INTEGRATE defines input features for a GCN model by counting the number of satisfactions of these rules for each candidate edge or node class. Figure 1 explains the problem addressed by INTEGRATE through the example of a co-authorship network. We make the following key contributions: **(1)** We introduce INTEGRATE, the first graph feature representation capable of exploiting symmetries in complex, multi-relational data using first-order logic rules. **(2)** Going beyond the use of carefully designed hand-crafted rules, our method learns rules automatically to construct GCNs. This method for constructing graph features is also more sample-efficient than node embedding learning methods, allowing GCNs to be learned from *smaller data sets*. **(3)** Unlike most graph representation learning methods, INTEGRATE is not limited to simple (binary) edges. It naturally handles hyperedges by representing the graph in terms of first-order logical rules.

## 2 Background

*Graphs.* Many real-world domains are naturally represented as graphs [50] such as social networks [29], chemical interactions [11], biological systems [43], knowledge bases [3], and recommendation systems [28]. Each graph consists of a set of vertices or nodes and a set of edges between them. A graph $G$ is formally defined as $\langle V, E\rangle$ where $V$ and $E$ are the vertex and edge sets, respectively. Each edge in the graph connects two nodes and might be represented as a tuple $(u, v) \in E$ where $u, v \in V^2$. Many real-world domains require two extensions to this basic graph data structure: node and edge types, and edges connecting more than two nodes. Heterogeneous graphs extend basic graphs to include node and edge types. Such graphs can be represented as $\langle V, E, f_V, f_E\rangle$ where $f_V$ and $f_E$ are functions mapping each node $v \in V$ and each edge $e \in E$ to their corresponding types $t_v \in T_V$ and $t_e \in T_E$, respectively. Hypergraphs extend graphs to allow edges connecting more than two nodes; each hyperedge is represented as $(u_1, \ldots, u_n)$ where $u_1, \ldots, u_n \in V^n$. This structure is, clearly, not directly reducible to tabular form, making learning from such data and reasoning about the information contained in them challenging [15]. This has motivated a long line of AI research from rule-based approaches [42] to deep representation learning [21].

*Graph Convolutional Networks (GCNs).* Graph Convolutional Networks (GCNs, [21]) are a class of deep representation learning models that extend convolutional neural networks (CNNs) to handle graph-structured data. GCNs exploit local structure to learn representations for each node in the input graph by aggregating information from its neighbors. Concretely, a GCN's input represents the graph's information in two forms: node feature descriptions ($\phi(v)$) and node neighborhood structure (captured through the adjacency matrix $A$ of the graph). As a result, learning a useful GCN from a graph requires informative node feature representations.

*Constructing GCNs using graph embedding methods.* One way to learn node features is through graph embedding methods [6, 49]. Among these, translational distance models minimize the distance between nodes and edges under specific constraints or regularizing factors; examples include TransE [4] and KG2E [14]. Extending these approaches, newer methods embed graphs into more complex spaces, such as hyperbolic space [2, 25] and hypercomplex

[1]The code can be found at: https://github.com/ddhami/RD2GCN/.

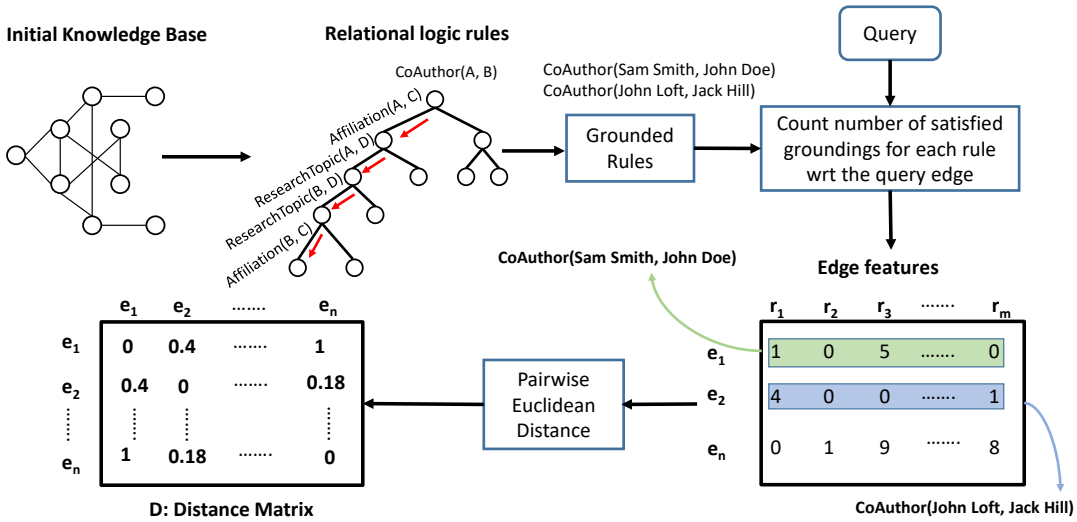


**Figure 2: Flowchart explaining graph feature construction using INTEGRATE. For the coauthorship link prediction task, we construct input features for a GCN in two steps. (1) We first learn a set of $m$ first-order logic rules to predict the presence of coauthorship between two authors. We use these rules to define a feature vector for each candidate edge. For each edge, for each rule, we count the number of ways that we can infer the presence of the edge by unifying the rule with facts based on the observed edges. (2) We use these edge feature vectors to construct a matrix representing the Euclidean distance between pairs of edges.**

space [44, 55]. Another important class of methods focuses on compositional operators for the graph's nodes and edges; examples include RESCAL [33], DistMult [52], and TuckER [1].

While these embedding methods are highly scalable, they face several issues in smaller but complex domains. First, they cannot represent hypergraphs. In practice, hyperedges are converted to a set of simple edges and additional nodes. However, learning embeddings for these new nodes becomes difficult since they are associated with very few edges. Second, these methods fail to capture complex patterns such as multi-hop interactions and aggregations. For instance, in the coauthorship network presented in Figure 1, embeddings cannot capture the pattern that two authors are more likely to be coauthors if they share a larger number of research areas. Embedding methods also struggle with generalization to new entities, such as new authors and research areas (cold-start problem). Finally, the embeddings learned by these methods are opaque, limiting their real-world applicability in high-stakes situations [39].

*Inducing meaningful patterns using Statistical Relational Learning.* One way to learn interpretable predictive models from graph-structured data is Statistical Relational Learning (SRL, [24, 37]). These methods compactly represent distributions over structured spaces by extending probabilistic graphical models (PGMs [23]) with first-order logic. They represent data in terms of logical predicates, allowing them to naturally represent heterogeneous graphs and hypergraphs by mapping each node class to unary predicates, each node's class label (say, node $v$ belonging to class $h$) to a logical atom based on the corresponding unary predicate ($h(v)$), each hyperedge type (say $t$) to an n-ary predicate, and each typed hyperedge ($u_1, \ldots, u_n$ of type $t$) to a logical atom based of the corresponding n-ary predicate ($t(u_1, \ldots, u_n)$). In this representation, node classification and link prediction problems are equivalent to predicting the conditional probability of a new predicate given all the predicates entailed by the graph.

These rules might be learned in a discriminative learning setup, consisting of a set of positive and negative examples. However, graphs typically do not contain negative instances of any edge type. One way to address this issue is to generate negative instances by randomly sampling from missing edges. This is equivalent to making the *closed-world assumption*, that is, that all unknown statements are false. However, this can result in sparse and imbalanced data sets, making the learning problem challenging. Another way is to reformulate the learning problem as a relational one-class classification problem [20]. These methods learn rules from only positive examples to estimate the density of new instances of the target predicate in terms of their distance to known instances.

For example, in the network presented in Figure 1, a first-order logic rule to predict coauthorship could capture the pattern that coauthorship is likely between authors who share a research area: `coauthor(A, B)` ⟵ `area(A, C)` ∧ `area(B, C)`. However, it cannot capture patterns about aggregates, such as the number of research areas common to two authors. [17] address this issue by learning a predictive model using the number of satisfactions of the body of the rules. So, for a candidate edge `couathor("Sam Smith", "Jane Doe")`, we can capture information about the number of shared research areas by counting the number of satisfactions of the rule.

*Related work.* Constructing feature representations from graphs is a prerequisite for effective graph analytics. As a result, researchers have developed variations of the standard GCN to better represent complex graphs. Relational GCNs [40] use relation-specific transformations to ensure the node representations capture different kinds of relationships between entities. CompGCNs [47] jointly embeds both nodes and relations in a graph. Graph Attention Networks (GAT, [48]) do not use convolutions but masked attention; this allows each node to dynamically weigh the importance of its neighbors' features when aggregating information.

## 3 INTEGRATE

Graph features are necessary for deep representation learning methods such as GCNs. Unfortunately, existing graph embedding methods not only yield opaque representations but also fail to capture the symmetries in multi-relational data. This limits the applicability of GCNs to real-world, high-stakes, small but complex graphs. Concretely, we aim to solve the following:

**Given:** An incomplete heterogeneous hypergraph $G$, consisting of nodes $V$ and hyperedges $E$, where each node belongs to a type in $T_V$, each hyperedge belongs to a type in $T_E$, and functions $f_V$ and $f_E$ map nodes and hyperedges to their respective types.
**To Do:** Construct interpretable yet effective input features for a GCN model to accurately complete $G$ by classifying unlabeled edges based on whether they belong to a target type $t \in T_E$ or classify unlabeled nodes based on whether they belong to a target class $t \in T_V$.

We address this problem via Statistical Relational Learning. To this effect, we represent $G$ as a knowledge base in first-order logic, consisting of a set of logical constants, one for each node in $V$; a set of logical predicates $T$, one for each node class in $T_V$ and each hyperedge type in $T_E$; and a set of logical atoms $\mathcal{A}$, obtained by applying the constants to the predicates, each atom representing either an observed typed hyperedge $e \in E$ or the assignment of a class to a node. In this representation, node classification is equivalent to predicting the unary predicate $t \in T_V$, representing the target node class, instantiated for each node $v$ as $t(v)$; link prediction is equivalent to predicting the n-ary predicate $t \in T_E$, representing the target hyperedge type, instantiated for each candidate hyperedge $(u_1, \ldots, u_n)$ as $t(u_1, \ldots, u_n)$. Using this representation of $G$, we first learn a set of first-order logic rules (say $R$) to predict predicate $t$ and then use the rules to define an input representation for the GCN model. We hypothesize that (and as our empirical results demonstrate), these features effectively capture *rich, higher-order information* about each node's attributes and its relationships, allowing the resulting GCN model to accurately complete the graph $G$ by predicting missing predicates (*i.e.,* typed nodes and edges). The overall approach is presented in Fig. 2.

*Constructing rich yet meaningful graph features.* The first step in our method is learning a set of rules, $R$, to predict instances of predicate $t$. We learn $k$ rules predicting the presence of the target predicate, $R^+$, and $k$ rules predicting its absence, $R^-$. To learn the rules to predict the presence of target predicate, $R^+ = \{r_1, \ldots, r_k\}$, we learn a relational one-class classifier [20] on the observed predicates. We learn rules for predicting the absence of the target, $R^-$, by generating examples randomly sampled from missing predicates. The full set of rules is defined as $R = R^+ \cup R^-$.

We use these rules to define features for each instance of the target predicate. For each instance $t(u_1, \ldots, u_n)$, we define the feature vector $\phi(t(u_1, \ldots, u_n))$ of length $2k$ where the $ith$ element corresponds to the number of ways that the rule $r_i$ can be true, when its head is unified with $t(u_1, \ldots, u_n)$. Concretely, $\phi(t(u_1, \ldots, u_n))_i = \text{Count}(t(u_1, \ldots, u_n), r_i)\ \forall i = 1, \ldots, 2k$. Here, Count is a function that unifies $r_i$'s head with $t(u_1, \ldots, u_n)$, retrieves all the atoms or their conjunctions that can be unified with the partially grounded rule to make it fully grounded, and returns the total number of such groundings.

**Example.** For the coauthorship network, consider two rules $r_1$ and $r_2$, one for identifying positive examples and the other for negative examples.
$r_1$ : `coauthor(A, B) ← area(A, C), area(B, C).`
$r_2$ : `coauthor(A, B) ← affil(A, C), affil(B, D), conflict(C,D).`
Then, for the query `coauthor("Sam Smith", "Jane Doe")`, the feature vector $[1, 0]$ constructed using $(r_1, r_2)$ represents the fact that the two authors share one research area and have no affiliations that conflict with each other.

*Constructing an edge distance matrix.* The original GCN formulation [21] requires a node-adjacency matrix $A$ to perform the layer-wise propagation. Instead of building the adjacency matrix from the graph's relations, we compute an edge-distance matrix [5, 38] $\mathcal{D}$ using our edge feature representations. This matrix approximates the adjacency matrix and, as our experiments show, captures richer structural information from the graph. In order to obtain $\mathcal{D}$, a pairwise Euclidean distance of all the node feature descriptors i.e. the counts in the feature set is computed. Algorithm 1 demonstrates the overall approach.

## 4 Experimental Evaluation

| Name | #Entities | #Preds | #Pos | #Neg | #Facts | #Rules |
|---|---|---|---|---|---|---|
| ICML | 6,653 | 4 | 155 | 6,498 | 1,395 | 7 |
| ICLR | 10,990 | 4 | 990 | 10,000 | 4,730 | 7 |
| DDI | 18,060 | 14 | 2,832 | 3,188 | 1,774 | 25 |
| Carcino | 440 | 8 | 182 | 258 | 54,890 | 9 |
| PPMI | 1,190 | 38 | 378 | 812 | 314,144 | 18 |
| CiteSeer | 15,008 | 17 | 7,504 | 7,504 | 119,635 | 7 |
| WebKB | 746 | 5 | 153 | 593 | 1,354 | 7 |

**Table 1: Summary statistics of the 7 data sets used for our empirical evaluation; the first 3 for the link prediction and the remaining 4 for the node classification. For each data set, we present the number of entities, predicates, positive and negative examples, background facts, and rules learned by our method. The last column lists the number of rules learned for each dataset. Of these data sets, only WebKB has binary predicates while the rest of the data sets have predicates of varying arity (i.e., hyperedges).**

Through our extensive experimental evaluation, we aim to answer the following research questions: **(Q1)** Can INTEGRATE effectively exploit the multi-relational structure of the graph to learn

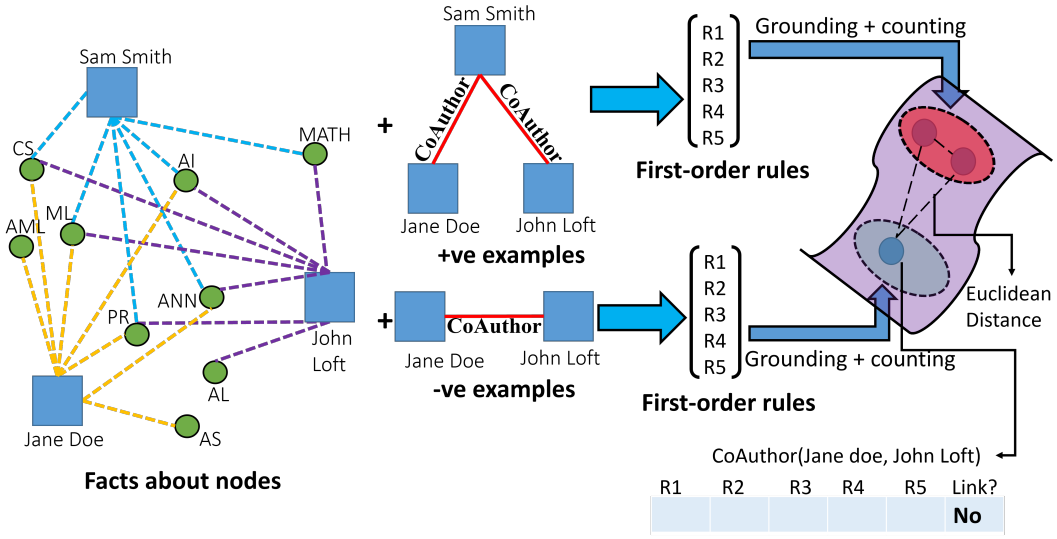


**Figure 3: INTEGRATE learns class-specific relational density estimators. For e.g., we use the co-authorship network on the left to learn separate rule sets for positive and negative examples. This method results in edge features that are more discriminative.**

+ Interacts($d_1$, $d_2$) $\implies$ TransporterSubstrate($d_1$, $tr_1$) $\wedge$ TransporterSubstrate($d_2$, $tr_1$) $\wedge$ EnzymeInhibitor($d_1$, $e_1$) $\wedge$ EnzymeInhibitor($d_2$, $e_1$)
+ Interacts($d_1$, $d_2$) $\implies$ EnzymeInducer($d_1$, $e_1$) $\wedge$ EnzymeSubstrate($d_2$, $e_1$) $\wedge$ EnzymeInducer($d_2$, $e_2$) $\wedge$ EnzymeInducer($d_1$, $e_2$)

- Interacts($d_1$, $d_2$) $\implies$ TargetInhibitor($d_1$, $t_1$) $\wedge$ TargetInhibitor($d_2$, $t_2$) $\wedge$ TransporterSubstrate($d_1$, $tr_1$)
- Interacts($d_1$, $d_2$) $\implies$ TargetAgonist($d_1$, $t_1$) $\wedge$ TargetAgonist($d_2$, $t_2$) $\wedge$ Transpor terInducer($d_1$, $tr_1$) $\wedge$ TransporterInducer($d_2$, $tr_2$)

**Figure 4: Sample rules learned from the drug-drug interactions data set. Positive rules are labeled with a '+' while negative rules are labeled with a '-'.**

+ Carcino($d$) $\implies$ drugAtom($d$,$a_1$) $\wedge$ sbond7($d$,$a_1$,$a_2$) $\wedge$ sbond1($d$,$a_2$,$a_3$) $\wedge$ sbond2($d$,$a_3$,$a_4$) $\wedge$ sbond1($d$,$a_4$,$a_5$) $\wedge$ sbond1($d$,$a_5$,_)
+ Carcino($d$) $\implies$ drugAtom($d$,$a_1$) $\wedge$ sbond7($d$, $a_1$, $a_2$) $\wedge$ sbond1($d$, $a_2$, $a_3$) $\wedge$ sbond2($d$, $a_3$, _))

- Carcino($d$) $\implies$ drugAtom($d$, $a_1$), sbond2($d$, $a_1$, $a_2$), sbond1($d$, $a_1$, _), sbond1($d$, $a_2$, _))
- Carcino($d$) $\implies$ drugAtom($d$, $a_1$), sbond7($d$, $a_1$, _)

**Figure 5: Sample rules learned from the carcinogenesis data set. Positive rules are labeled with a '+' while negative rules are labeled with a '-'.**

rich representations from small data sets? **(Q2)** Does relational density estimation yield better rules in imbalanced data sets? **(Q3)** Is constructing a distance-based graph structure useful? **(Q4)** How does the choice of distance measures affect the performance of INTEGRATE?

*Data sets.* To answer these research questions, we consider*seven* data sets from the statistical relational learning community for two kinds of tasks – link prediction and node classification. For link prediction, we consider *three* knowledge graph data sets – ICML, ICLR, and DDI. The ICML, ICLR data sets consist of information about papers and their authors from the eponymous conferences; this information was extracted from the Microsoft Academic Graph (MAG, [41]). Here, the task is to predict the coauthorship relation between two authors. The DDI data set contains information about drugs and their interactions [11], and the link to be predicted is whether two drugs interact. For node classification, we consider *three* data sets – Carcino, PPMI, CiteSeer, WebKB. *Carcino* is a biomedical data set of the structures of chemical compounds; the task is to predict if they are carcinogenic. *PPMI* consists of data from study [30] designed to identify biomarkers that impact Parkinson's disease; the task is to predict if a patient has Parkinson's [12]. *CiteSeer* is a relational data set of citations [35]; the task is to predict the author of a citation. *WebKB* consists of web pages and hyperlinks from 4 CS departments [7]; the task is to predict if someone is a faculty member. Table 1 summarizes the seven data sets.

**Algorithm 1** INTEGRATE: Feature and Structure Construction for GCN

**Input**
  $\mathcal{K}$ Knowledge Base, representing graph $G$
  $t$ Target predicate, representing a node or edge type
  k Number of rules to learn per class
**Output**
  $R$ Rule set used to define features
  $X$ Feature matrix
  $\mathcal{D}$ Distance matrix (to replace Adjacency Matrix)
// **Phase 1: Rule Learning**
$P_{\text{obs}} \leftarrow$ Extract observed instances of $t$ from $\mathcal{K}$
$R^+ \leftarrow \textsc{LearnRelOCC}(P_{\text{obs}}, k)$ ▷ Learn $k$ rules for presence
$N_{\text{samp}} \leftarrow \textsc{RandomSampleMissing}(t, \mathcal{K})$ ▷ Sample negative examples
$R^- \leftarrow \textsc{LearnRelOCC}(N_{\text{samp}}, k)$ ▷ Learn $k$ rules for absence
$R \leftarrow R^+ \cup R^-$ ▷ Total set of $2k$ rules
// **Phase 2: Feature Construction (Counting Satisfactions)**
$I_{\text{cand}} \leftarrow \textsc{GetAllCandidateInstances}(t, \mathcal{K})$ ▷ All nodes or edges to classify
Initialize $X$ as empty matrix of size $|I_{\text{cand}}| \times |R|$
**for** each candidate instance $u \in I_{\text{cand}}$ **do**
  **for** each rule $r_i \in R$ where $i \in \{1, \ldots, 2k\}$ **do**
    $\theta \leftarrow$ Unify head of $r_i$ with atom $u$
    $count \leftarrow \textsc{CountGroundings}$(body of $r_i$, $\theta$, $\mathcal{K}$) ▷ Count satisfied groundings
    $X[u][i] \leftarrow count$
  **end for**
**end for**
// **Phase 3: Distance Matrix Construction**
Initialize $\mathcal{D}$ as matrix of size $|I_{\text{cand}}| \times |I_{\text{cand}}|$
$D \leftarrow \textsc{PairwiseEuclideanDistance}(X)$ ▷ Compute distance between feature vectors
**return** $R, X, \mathcal{D}$

*Methods.* We compare our proposed framework, INTEGRATE, against *four* kinds of baselines: relational graph embedding methods, rule-learning methods, GCN-based methods, and SRL methods. Under relational graph embedding methods, we consider *six* baselines: ComplEx [46], ConvE [10], SimplE [18], ReInceptionE [51], the Functional variant of ExpressiveE [34], and HousE+ [27]. Under rule-based methods, we consider *four* baselines: Handwritten rules, rules learned using a differentiable logic system (Neural-LP, [53]), rules constructed from random walks based on user-defined metapaths (metapath2vec, [13]), rules constructed from Path ranking-based random walks (PRA, [16, 26]), and a logistic regression model (LR) over propositional features learned using the **Node+LinkFeat** algorithm (N+LF, [45]). Under GCN methods, we consider*four* baselines: R-GCN [40], CompGCN [47] that jointly embeds both nodes and relations in a graph, NBFNet [56] that uses the generalized Bellman-Ford algorithm for link prediction, and SEAL [54] that extracts a local subgraph around each target link. Under SRL methods, we consider *two* baselines: MLN-Boost [19] and RDN-Boost [32]. Apart from these, we also compare our methods with GATs [48] and GCNs with the Heat Diffusion (HD) and Personalized Page Rank (PPR) kernels [22].

| Method | ICML | | ICLR | | DDI | |
|---|---|---|---|---|---|---|
| | F1 | APR | F1 | APR | F1 | APR |
| ComplEx | 0.03 | 0.04 | 0.06 | 0.11 | 0.62 | 0.71 |
| ConvE | 0.02 | 0.02 | 0.07 | 0.05 | 0.54 | 0.68 |
| SimplE | 0.02 | 0.13 | 0.10 | 0.54 | 0.45 | 0.50 |
| ReInceptionE | 0.03 | 0.14 | 0.07 | 0.08 | 0.53 | 0.83 |
| ExpressiveE-F | 0.02 | 0.02 | 0.12 | 0.95 | 0.56 | 0.91 |
| HouseE+ | 0.03 | 0.56 | 0.11 | 0.77 | 0.53 | 0.84 |
| Handwritten | 0.17 | 0.13 | 0.66 | 0.50 | 0.56 | 0.58 |
| Neural-LP$_3$ | 0.05 | 0.27 | 0.46 | 0.42 | 0.46 | 0.37 |
| metapath2vec | 0.34 | 0.29 | 0.48 | 0.64 | 0.71 | 0.71 |
| PRA | 0.0 | 0.51 | 0.0 | 0.54 | 0.53 | 0.70 |
| N + LF (LR) | 0.55 | 0.40 | 1.0 | 0.98 | 0.79 | 0.78 |
| R-GCN | 0.13 | 0.13 | 0.72 | 0.76 | 0.73 | 0.92 |
| CompGCN | 0.04 | 0.19 | **0.80** | 0.91 | 0.68 | 0.83 |
| NBFNet | 0.06 | 0.76 | 0.64 | **0.99** | 0.62 | 0.87 |
| SEAL | 0.20 | **0.78** | 0.67 | 0.92 | 0.59 | 0.85 |
| INTEGRATE | **0.56** | 0.56 | 0.75 | 0.97 | **0.99** | **1.0** |

**Table 2: Link prediction. Performance of our method, INTEGRATE, on the task of link prediction in terms of F1 score (F1) and AUC-PR (APR), as compared to 6 graph-embedding, 5 rule-based, and 4 GCN-based methods.**

*Metrics.* We compare our method, INTEGRATE, against the baselines on link prediction and node classification tasks by quantifying each method's performance using *two* metrics – F1-score and the area under the precision-recall curve (AUC-PR).

*Setup.* The INTEGRATE model consists of a GCN with 2 hidden layers, each of dimension 16, with a dropout layer between the 2 graph convolutional layers. We introduce non-linearity between input and hidden layers using the *ReLU* function; we compute prediction scores using the softmax function. To make predictions using the scores, we compute the threshold as the average of the obtained scores in the test set.[2] Finally, note that many of the baselines assume binary edges. However, out of our *six* data sets, only the WebKB data set's edges are binary. All other data sets have some hyperedges. So, if a baseline cannot handle hyperedges, we convert them to $\binom{n}{2}$ binary edges.

*Results.* We can now answer our research questions.

**(Q1)** We address this question in two parts based on the two tasks considered:
**Part 1: Link prediction.** Table 2 compares the performance of INTEGRATE on the link prediction task with two deep learning-based approaches: graph embedding and GCN-based methods. Our method outperforms all the baselines significantly in 2 of the 3 data sets, with the difference being significant in the smaller *ICML* data set. In the 3rd data set, INTEGRATE's performance is comparable to the baselines. This demonstrates that INTEGRATE is significantly better than the strong baselines for this task.

[2]We run our experiments on a machine with 8 GeForce GTX 1080 Ti cards.

| Method | Carcino | | PPMI | | CiteSeer | | WebKB | |
|---|---|---|---|---|---|---|---|---|
| | F1 | APR | F1 | APR | F1 | APR | F1 | APR |
| Neural-LP$_3$ | 0.13 | 0.13 | 0.0 | 0.56 | 0.0 | 0.62 | 0.0 | 0.53 |
| metapath2vec | 0.41 | 0.36 | 0.65 | 0.79 | 0.91 | 0.98 | 0.28 | 0.19 |
| PRA | 0.0 | 0.50 | 0.43 | 0.62 | 0.66 | 0.75 | 0.47 | 0.70 |
| N+LF (LR) | 0.69 | 0.73 | 0.52 | 0.57 | 0.73 | 0.64 | 0.53 | 0.48 |
| R-GCN | 0.39 | 0.57 | 0.74 | 0.73 | **0.96** | **0.99** | 0.21 | 0.25 |
| MLNB | 0.34 | 0.30 | 0.80 | **0.97** | **0.96** | 0.98 | **1.0** | **1.0** |
| RDNB | 0.27 | 0.20 | **0.85** | 0.95 | 0.95 | 0.98 | **1.0** | **1.0** |
| INTEGRATE | **0.79** | **0.93** | 0.61 | 0.80 | 0.73 | 0.82 | 0.36 | 0.61 |

**Table 3: Node classification. Performance of our method, INTEGRATE, on the task of link prediction in terms of F1 score (F1) and AUC-PR (APR), as compared to 4 rule-based methods, R-GCN, and SRL methods**

**Part 2: Node classification.** Table 3 compares the performance of INTEGRATE on the node classification task with GCN and SRL-based methods, respectively. INTEGRATE outperforms the SRL and GCN baselines in the smaller *Carcino* and *PPMI* data sets while maintaining comparable performance in the others. Hence, INTEGRATE yields models that are effective at node classification, especially in small data sets. Rules learned from the DDI and Carcino data sets are shown in Figures 4 and 5. Thus, our evaluations on both tasks support our hypothesis that *rule-based embeddings capture richer abstract features*, yielding more predictive models. The results demonstrate that our method significantly outperforms GCN baselines, especially in the smaller domains. On these domains, INTEGRATE also outperforms more powerful diffusion-based GCNs ([22]; table 4). Thus, we can answer **Q1** affirmatively. Further ablation studies are presented in Tables 5 and 6.

**(Q2)** To evaluate the efficacy of relational density estimation in learning discriminative rules from small graph data sets, we compare INTEGRATE with *four* rule learning methods: handwritten rules (Gaifman), NeuralLP[3], metapath2vec and PRA. Tables 2 and 3 demonstrate that using our *density estimation method significantly outperforms existing rule-based methods across all domains*. Of these, PRAGCN is an especially interesting baseline since its performance drops significantly in the highly imbalanced domains. In the imbalanced domain from link prediction domains (*ICML* and *ICLR*), PRA learns features that classify all examples as negative. In the imbalanced node classification domains (*Carcino* and *CiteSeer*), PRA features classify *all examples* as positive, thus being biased towards a single (the larger) density. In contrast, INTEGRATE's relational density estimation-based rules allow it to better capture both positive and negative classes, allowing it to achieve high performance in these challenging domains. This answers **Q2** affirmatively.

**(Q3)** The main advantage of our method is learning a secondary graph structure where both link prediction and node classification tasks become simple prediction tasks. As can be seen from the results for link prediction, a simple discriminative machine learning algorithm (logistic regression), used on top of the learned features (N+LF), performs better than the other baselines, including GCN-based. In the case of node classification, the results are comparable.

To show the importance of using a distance matrix, we compare our method with Graph Attention Networks (GATs, [48]). Figure 6 shows that using a distance matrix can be an effective alternative to the original adjacency matrix. We also compare our approach with variations of GAT models augmented with PRA-based distance matrices. These distance matrices yield worse models (Fig. 7). Hence, not only does INTEGRATE yield a richer feature representation, but the distance matrices also effectively capture node relationships; this answers **Q3** affirmatively.

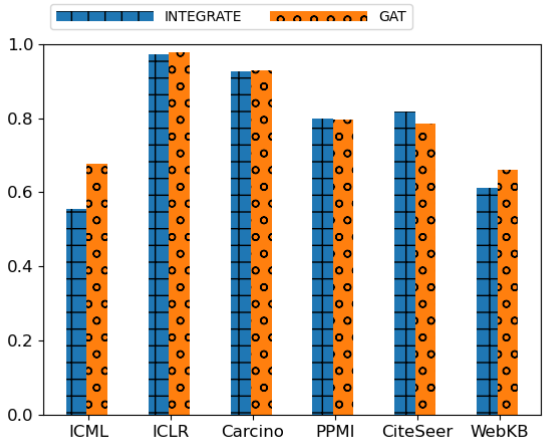


**Figure 6: Comparison (AUC-PR) with GATs showing the importance of node-distance matrix $\mathcal{D}$.**

**(Q4)** Figure 8 presents the effect of distance measures on the performance of INTEGRATE on the DDI data set. We compare INTEGRATE's Euclidean distance matrix with equivalent models using Manhattan ($L_1$) and Chebyshev ($L_\infty$) distances. Since Euclidean is the shortest distance between nodes, as expected, it performs the best. This answers **Q4**.

[3]We consider NeuralLP with rules of length 3. Longer rule length has similar performance (see appendix).

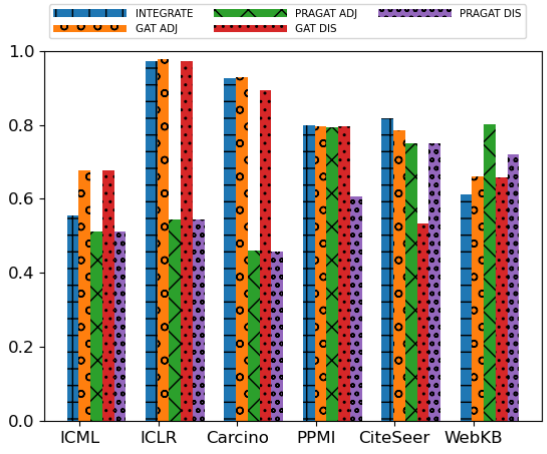


**Figure 7: AUC-PR comparison with variations of GAT with PRA features, distance matrix, and distance matrix as adjacency matrix.**

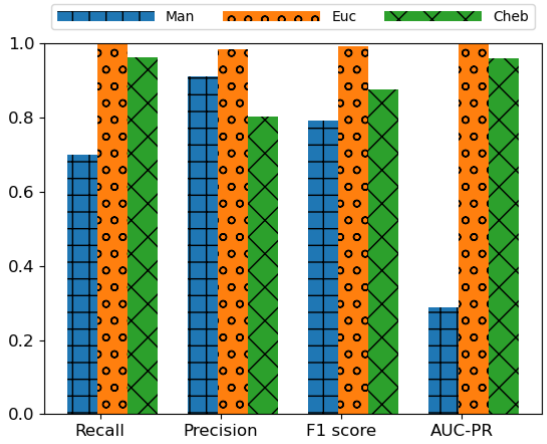


**Figure 8: Link prediction performance on the DDI data set using INTEGRATE across different choices of distance measures (Manhattan, Euclidean, and Chebyshev) used to construct the distance matrix.**

| Data | Method | Recall | Precision | F1 | AUC-PR |
|---|---|---|---|---|---|
| *ICML* | HD + GCN | 0.35 | **1.0** | 0.52 | 0.54 |
| | PPR + GCN | 0.35 | **1.0** | 0.52 | 0.54 |
| | INTEGRATE | **0.37** | **1.0** | **0.54** | **0.69** |
| *Carcino* | HD + GCN | 0.60 | **0.97** | 0.74 | 0.90 |
| | PPR + GCN | 0.61 | 0.92 | 0.73 | 0.88 |
| | INTEGRATE | **0.66** | **0.97** | **0.79** | **0.93** |

**Table 4: Comparison of the performance of INTEGRATE with diffusion-based models on a link prediction domain (ICML) and a node classification domain (Carcino)**

**Table 5: Effect of size of hidden layers.**

| Data | Size | Recall | Precision | F1 Score | AUC-PR |
|---|---|---|---|---|---|
| *ICML'18* | 32 | 0.37 | 1.0 | 0.54 | 0.69 |
| | 64 | 0.37 | 1.0 | 0.54 | 0.69 |
| | 128 | 0.37 | 1.0 | 0.54 | 0.69 |
| *DDI* | 32 | 0.99 | 1.0 | 0.99 | 1.0 |
| | 64 | 0.99 | 1.0 | 0.99 | 1.0 |
| | 128 | 0.99 | 1.0 | 0.99 | 1.0 |

**Table 6: Effect of number of hidden layers.**

| Data | # | Recall | Precision | F1 Score | AUC-PR |
|---|---|---|---|---|---|
| *ICML'18* | 3 | 0.369 | 1.0 | 0.539 | 0.692 |
| | 4 | 0.369 | 1.0 | 0.539 | 0.692 |
| | 5 | 0.369 | 1.0 | 0.539 | 0.692 |
| *DDI* | 3 | 0.989 | 0.998 | 0.994 | 0.999 |
| | 4 | 1.0 | 0.997 | 0.998 | 0.999 |
| | 5 | 0.999 | 0.991 | 0.995 | 0.997 |

## 5 Conclusion and outlooks

We presented a method to construct rich yet interpretable graph embeddings. These embeddings outperform existing methods, especially on small multi-relational data. There are several directions for future research. The first is to allow for joint learning and inference over multiple predicates. Next, instead of a relational density estimator, one could use more classical rule learning techniques [31, 36, 42] to learn the rules. Finally, our approach could be made more scalable by integrating sampling and approximate counting methods [8], reducing the learning time considerably without sacrificing performance.

## 6 Acknowledgements

The authors gratefully acknowledge the support of the AFOSR award FA9550-23-1-0239, the ARO award W911NF2010224 and the DARPA Assured Neuro Symbolic Learning and Reasoning (ANSR) award HR001122S0039. The Eindhoven University of Technology authors received support from their Department of Mathematics and Computer Science and the Eindhoven Artificial Intelligence Systems Institute.

## References

[1] Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. TuckER: Tensor Factorization for Knowledge Graph Completion. 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)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 5185–5194. doi:10.18653/v1/D19-1522

[2] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. Multi-relational Poincaré Graph Embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 4465–4475. https://proceedings.neurips.cc/paper/2019/hash/f8b932c70d0b2e6bf071729a4fa68dfc-Abstract.html

[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

[4] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 2787–2795. https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html

[5] Andries E Brouwer and Willem H Haemers. 2011. *Spectra of graphs*. Springer Science & Business Media.

[6] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering* 30, 9 (2018), 1616–1637.

[7] Mark Craven, Andrew McCallum, Dan PiPasquo, Tom Mitchell, and Dayne Freitag. 1998. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[8] Mayukh Das, Devendra Singh Dhami, Gautam Kunapuli, Kristian Kersting, and Sriraam Natarajan. 2019. Fast Relational Probabilistic Inference and Learning: Approximate Counting via Hypergraphs. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 7816–7824. doi:10.1609/aaai.v33i01.33017816

[9] John S. Denker, W. R. Gardner, Hans Peter Graf, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, Lawrence D. Jackel, Henry S. Baird, and Isabelle Guyon. 1988. Neural Network Recognizer for Hand-Written Zip Code Digits. In *NeurIPS*. Morgan Kaufmann, 323–331.

[10] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 1811–1818. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17366

[11] Devendra Singh Dhami, Gautam Kunapuli, Mayukh Das, David Page, and Sriraam Natarajan. 2018. Drug-drug interaction discovery: kernel learning from heterogeneous similarities. *Smart Health* 9 (2018), 88–100.

[12] Devendra Singh Dhami, Ameet Soni, David Page, and Sriraam Natarajan. 2017. Identifying parkinson's patients: A functional gradient boosting approach. In *Conference on Artificial Intelligence in Medicine in Europe*. Springer, 332–337.

[13] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. ACM, 135–144. doi:10.1145/3097983.3098036

[14] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. 2015. Learning to Represent Knowledge Graphs with Gaussian Embedding. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu (Eds.). ACM, 623–632. doi:10.1145/2806416.2806502

[15] David D. Jensen and Jennifer Neville. 2002. Linkage and Autocorrelation Cause Feature Selection Bias in Relational Learning. In *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002*, Claude Sammut and Achim G. Hoffmann (Eds.). Morgan Kaufmann, 259–266.

[16] Navdeep Kaur, Gautam Kunapuli, Saket Joshi, Kristian Kersting, and Sriraam Natarajan. 2019. Neural networks for relational data. In *International Conference on Inductive Logic Programming*. Springer, 62–71.

[17] Seyed Mehran Kazemi, David Buchman, Kristian Kersting, Sriraam Natarajan, and David Poole. 2014. Relational Logistic Regression. In *KR*. AAAI Press.

[18] Seyed Mehran Kazemi and David Poole. 2018. SimplE Embedding for Link Prediction in Knowledge Graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 4289–4300. https://proceedings.neurips.cc/paper/2018/hash/b2ab001909a8a6f04b51920306046ce5-Abstract.html

[19] Tushar Khot, Sriraam Natarajan, Kristian Kersting, and Jude Shavlik. 2011. Learning markov logic networks via functional gradient boosting. In *2011 IEEE 11th international conference on data mining*. IEEE, 320–329.

[20] Tushar Khot, Sriraam Natarajan, and Jude W. Shavlik. 2014. Relational One-Class Classification: A Non-Parametric Approach. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, Carla E. Brodley and Peter Stone (Eds.). AAAI Press, 2453–2459. http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8578

[21] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[22] Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. 2019. Diffusion Improves Graph Learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 13333–13345. https://proceedings.neurips.cc/paper/2019/hash/23c894276a2c5a16470e6a31f4618d73-Abstract.html

[23] Daphane Koller. 2009. Probabilistic Graphical Models: Principles and Techniques.

[24] Daphne Koller, Nir Friedman, Sašo Džeroski, Charles Sutton, Andrew McCallum, Avi Pfeffer, Pieter Abbeel, Ming-Fai Wong, Chris Meek, Jennifer Neville, et al. 2007. *Introduction to statistical relational learning*. MIT press.

[25] Prodromos Kolyvakis, Alexandros Kalousis, and Dimitris Kiritsis. 2020. HyperKG: Hyperbolic Knowledge Graph Embeddings for Knowledge Base Completion. *European Semantic Web Conference*.

[26] Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine learning* 81, 1 (2010), 53–67.

[27] Rui Li, Jianan Zhao, Chaozhuo Li, Di He, Yiqi Wang, Yuming Liu, Hao Sun, Senzhang Wang, Weiwei Deng, Yanming Shen, Xing Xie, and Qi Zhang. 2022. HousE: Knowledge Graph Embedding with Householder Parameterization. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.). PMLR, 13209–13224. https://proceedings.mlr.press/v162/li22ab.html

[28] Xin Li and Hsinchun Chen. 2013. Recommendation as link prediction in bipartite graphs: A graph kernel-based machine learning approach. *Decision Support Systems* 54, 2 (2013), 880–890.

[29] Li Liu, William K. Cheung, Xin Li, and Lejian Liao. 2016. Aligning Users across Social Networks Using Network Embedding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, Subbarao Kambhampati (Ed.). IJCAI/AAAI Press, 1774–1780. http://www.ijcai.org/Abstract/16/254

[30] Kenneth Marek, Danna Jennings, Shirley Lasch, Andrew Siderowf, Caroline Tanner, Tanya Simuni, Chris Coffey, Karl Kieburtz, Emily Flagg, Sohini Chowdhury, et al. 2011. The Parkinson progression marker initiative (PPMI). *Progress in neurobiology* 95, 4 (2011), 629–635.

[31] Stephen Muggleton. 1995. Inverse entailment and Progol. *New generation computing* 13, 3 (1995), 245–286.

[32] Sriraam Natarajan, Tushar Khot, Kristian Kersting, Bernd Gutmann, and Jude Shavlik. 2012. Gradient-based boosting for statistical relational learning: The relational dependency network case. *Machine Learning* 86, 1 (2012), 25–56.

[33] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, Lise Getoor and Tobias Scheffer (Eds.). Omnipress, 809–816. https://icml.cc/2011/papers/438_icmlpaper.pdf

[34] Aleksandar Pavlovic and Emanuel Sallinger. 2023. ExpressivE: A Spatio-Functional Embedding For Knowledge Graph Completion. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/pdf?id=xkev3_np08z

[35] Hoifung Poon and Pedro Domingos. 2007. Joint inference in information extraction. In *AAAI Conference on Artificial Intelligence*, Vol. 7. 913–918.

[36] J. Ross Quinlan. 1990. Learning logical definitions from relations. *Machine learning* 5, 3 (1990), 239–266.

[37] Luc De Raedt, Kristian Kersting, Sriraam Natarajan, and David Poole. 2016. *Statistical Relational Artificial Intelligence: Logic, Probability, and Computation*. Morgan & Claypool Publishers.

[38] Dennis H Rouvray and Alexandru T Balaban. 1979. Chemical applications of graph theory. *Applications of Graph Theory* 177 (1979), 155–156.

[39] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* 1, 5 (2019), 206–215.

[40] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. *European Semantic Web Conference* (2018).

[41] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*. 243–246.

[42] Ashwin Srinivasan. 2001. The aleph manual.

[43] Aravind Subramanian, Pablo Tamayo, Vamsi K Mootha, Sayan Mukherjee, Benjamin L Ebert, Michael A Gillette, Amanda Paulovich, Scott L Pomeroy, Todd R Golub, Eric S Lander, et al. 2005. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of*

*the National Academy of Sciences* 102, 43 (2005), 15545–15550.

[44] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=HkgEQnRqYQ

[45] Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, Alexandre Allauzen, Edward Grefenstette, Karl Moritz Hermann, Hugo Larochelle, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, Beijing, China, 57–66. doi:10.18653/v1/W15-4007

[46] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings, Vol. 48)*, Maria-Florina Balcan and Kilian Q. Weinberger (Eds.). JMLR.org, 2071–2080. http://proceedings.mlr.press/v48/trouillon16.html

[47] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based Multi-Relational Graph Convolutional Networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=BylA_C4tPr

[48] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=rJXMpikCZ

[49] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[50] Feng Xia, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. 2021. Graph learning: A survey. *IEEE Transactions on Artificial Intelligence* 2, 2 (2021), 109–127.

[51] Zhiwen Xie, Guangyou Zhou, Jin Liu, and Jimmy Xiangji Huang. 2020. ReInceptionE: Relation-Aware Inception Network with Joint Local-Global Structural Information for Knowledge Graph Embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 5929–5939. doi:10.18653/v1/2020.acl-main.526

[52] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6575

[53] Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 2319–2328. https://proceedings.neurips.cc/paper/2017/hash/0e55666a4ad822e0e34299df3591d979-Abstract.html

[54] Muhan Zhang and Yixin Chen. 2018. Link Prediction Based on Graph Neural Networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 5171–5181. https://proceedings.neurips.cc/paper/2018/hash/53f0d7c537d99b3824f0f99d62ea2428-Abstract.html

[55] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion Knowledge Graph Embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 2731–2741. https://proceedings.neurips.cc/paper/2019/hash/d961e9f236177d65d21100592edb0769-Abstract.html

[56] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal A. C. Xhonneux, and Jian Tang. 2021. Neural Bellman-Ford Networks: A General Graph Neural Network Framework for Link Prediction. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (Eds.). 29476–29490. https://proceedings.neurips.cc/paper/2021/hash/f6a673f09493afcd8b129a0bcf1cd5bc-Abstract.html

# A Additional figures and tables

**Table 7: Additional results for the link prediction task**

| Method | ICML | | ICLR | | DDI | |
|---|---|---|---|---|---|---|
| | F1 | APR | F1 | APR | F1 | APR |
| Neural-LP$_3$ | 0.05 | 0.27 | 0.46 | 0.42 | 0.46 | 0.37 |
| Neural-LP$_{10}$ | 0.07 | 0.14 | 0.43 | 0.45 | 0.47 | 0.4 |
| N+LF (LR) | 0.55 | 0.4 | 1 | 0.98 | 0.79 | 0.78 |
| N+LF (NN) | 0.56 | 0.41 | 0.56 | 0.41 | 0.82 | 0.83 |
| HouseE | 0.03 | 0.54 | 0.11 | 0.77 | 0.5 | 0.67 |
| HouseE+ | 0.04 | 0.56 | 0.11 | 0.77 | 0.53 | 0.84 |
| ExpressiveE-B | 0.02 | 0.02 | 0.11 | 0.96 | 0.55 | 0.88 |
| ExpressiveE-F | 0.02 | 0.02 | 0.12 | 0.95 | 0.56 | 0.91 |
| INTEGRATE | 0.56 | 0.56 | 0.75 | 0.97 | 0.99 | 1.0 |

**Table 8: Additional results for the node classification task**

| Method | Carcino | | PPMI | | CiteSeer | | WebKB | |
|---|---|---|---|---|---|---|---|---|
| | F1 | APR | F1 | APR | F1 | APR | F1 | APR |
| Neural-LP3 | 0.13 | 0.13 | 0 | 0.56 | 0 | 0.62 | 0 | 0.53 |
| Neural-LP10 | 0.21 | 0.16 | 0.38 | 0.28 | 0.32 | 0.44 | 0.12 | 0.01 |
| N+LF (LR) | 0.69 | 0.73 | 0.52 | 0.57 | 0.73 | 0.64 | 0.53 | 0.48 |
| N+LF (NN) | 0.7 | 0.54 | 0.51 | 0.34 | 0.85 | 0.78 | 0.57 | 0.4 |
| INTEGRATE | 0.79 | 0.93 | 0.61 | 0.80 | 0.73 | 0.82 | 0.36 | 0.61 |