

# Enhancing Graph Learning With First-Order Logic Rules

Ali Khazraee<sup>1</sup>, Abdolreza Mirzaei<sup>1,2</sup>, Majid Farhadi<sup>3</sup>, Parmis Naddaf<sup>1</sup>, Kiarash Zahirnia<sup>1</sup>, Mohammad Salameh<sup>4</sup>, Kevin Cannons<sup>4</sup>, Richard Mar<sup>1</sup>, Erfaneh Mahmoudzadeh<sup>1</sup>, Oliver Schulte<sup>1</sup>

<sup>1</sup>Simon Fraser University, <sup>2</sup>Isfahan University of Technology, <sup>3</sup>University of Alberta, <sup>4</sup>Huawei Technologies

khali7494@gmail.com, abdolreza\_mirzaei@sfu.ca, farhadi@ualberta.ca, parmis\_naddaf@sfu.ca, kiarash\_zahirnia@sfu.ca, mohammad.salameh@huawei.com, kevin.cannons@huawei.com, richard\_mar@sfu.ca, erfaneh\_mahmoudzadeh\_ahmadi\_nejad@sfu.ca, oschulte@cs.sfu.ca

## Abstract

Existing graph generative models produce graphs that are often quite realistic, but sometimes miss domain-specific patterns. Enhancing graph learning with domain knowledge is one of the current frontiers for neural models of graph data. In this paper, we propose a new approach to enhancing deep graph generative models with knowledge that is represented by first-order logic rules. First-order logic provides an expressive formalism for representing interpretable domain knowledge about relational structures. Our conceptual contribution is a new first-order semantic loss function for training a graph generative model on relational data: maximize the model likelihood subject to a *rule moment matching constraint*, namely that the expected instance count of each rule matches its observed instance count. Our algorithmic contribution is a novel method for computing the expected instance count of a first-order rule for a Variational Graph Autoencoder (VGAE), based on matrix multiplication. Empirical evaluation on seven benchmark datasets, both homogeneous and heterogeneous, shows that rule moment matching improves the quality of generated graphs substantially (by orders of magnitude on standard graph quality metrics), and improves predictive accuracy on the downstream task of node classification.

## 1 Introduction

Generative models for graphs based on graph neural networks (GNNs) have achieved great success in modeling complex graphs [Hamilton, 2020]. One of the current research frontiers is enhancing graph learning with domain knowledge [Tian *et al.*, 2024]. Different enhancement methodologies are appropriate for different types of knowledge. In this paper, we consider leveraging knowledge in the form of a *first-order logic knowledge base* [Russell and Norvig, 2010], comprising a set of if-then rules. An example rule would be “If person  $X$  works in city  $Y$ , then  $X$  lives in city  $Y$  (with probability  $p$ )”.

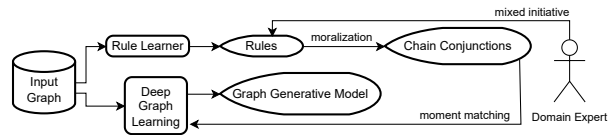


Figure 1: System Overview for Rule-Enhanced Graph Generation

**Advantages.** Logical rules have several advantages for enhancing graph learning. (1) *Expressiveness*: First-order rules are one of the most common formalisms for representing domain knowledge in AI and database systems [Russell and Norvig, 2010]. (2) *Interpretability*: If-then rules are easily understood by users and domain experts. In this work we derive rules from a first-order Bayesian network (FOBN) [Kimmig *et al.*, 2014; Wang *et al.*, 2008], which provides a graphical interpretation of associations among random variables. Under assumptions described in the causal discovery literature [Pearl, 2000; Spirtes *et al.*, 2000], the network links can be interpreted as causal relationships. (3) *Learnability*: The field of statistical-relational learning (SRL) has developed methods for learning a FOBN from a heterogeneous training graph, known as *structure learning*. (4) *User Control*: Users can control the behavior of the final graph generation system in a mixed-initiative approach, by specifying and/or rejecting rules. (5) *Graph Realism and Data Efficiency*: Matching first-order rules leads to generating more realistic graphs, while requiring less training data.

**Approach.** Figure 1 shows our system components. We show how fundamental ideas from SRL [Raedt *et al.*, 2016] can be combined with deep graph generative models (GGMs). A fundamental concept of SRL is *rule moment matching* [Domingos and Lowd, 2019; Russell, 2015; Kuzelka *et al.*, 2018]. The general idea is that a rule can be viewed as specifying a *motif* or subgraph pattern with an **instance count** in a given graph. Rule (moment) matching requires that *the expected rule instance count for a model should match the observed rule count in a training graph*. Our novel GGM training objective is to *maximize the GGM likelihood subject to rule matching*.

Our algorithmic contribution is a *differentiable new matrix multiplication method* for computing observed and expected

rule instance counts. We show that for every rule (satisfying a minor syntactic constraint), there is a corresponding sequence of adjacency matrices, such that i) the observed rule count is obtained by multiplying the data adjacency matrices, and ii) the expected rule count for a VGAE model is obtained by multiplying expected adjacency matrices.

**Evaluation.** Our methodology uses an A-B design where we compare training a recent state-of-the-art variational graph auto-encoder [Mahmoudzadeh *et al.*, 2024] with and without rule moment matching, on seven benchmark datasets. Our approach is in principle compatible with any FO rule learner; we deploy the Factorbase system, which uses BN graph structure learning to find a comprehensive set of first-order rules [Qian and Schulte, 2015]. We find that rule-enhanced VGAEs score better than standard VGAEs on several metrics: (1) They generate *more realistic graphs*, by orders of magnitude, as measured by SOTA graph quality metrics (F1 MMD) [Thompson *et al.*, 2022; O’Bray *et al.*, 2022]. (2) On the downstream task of *node classification*, the rule-enhanced VGAE node embeddings improve accuracy compared to standard VGAE. (3) Learning curves for node classification show that injecting domain knowledge with first-order rules often leads to more data efficient learning.

**Contributions** Our main contributions can be summarized as follows.

- A new semantic loss objective function for enhancing generative graph learning with domain knowledge represented by first-order logical rules.
- A new matrix multiplication algorithm for counting i) the number of rule instances in a graph and ii) the expected number of instances in an expected graph.
- A proof that the matrix multiplication algorithm can be used to estimate the expected number of rule instances for a Variational Graph Autoencoder model.
- Leveraging the matrix multiplication methods to compute the new semantic loss objective: Maximize the data likelihood of a graph generative model, constrained so that the observed number of rule instances matches the expected number of rule instances.

## 2 Related Work

Our work falls under the heading of *neuro-symbolic AI*, a cutting-edge field of AI that aims to combine symbolic formalisms, such as first-order logic, with neural network learning; see Figure 2. For surveys of neuro-symbolic AI, please see [Raedt *et al.*, 2020; Garcez and Lamb, 2023], and Kautz’s 2022 Englemore lecture. Within Kautz’s taxonomy, our approach belongs to the *semantic loss* frameworks (type 5) where symbolic knowledge is represented as a regularization term added to a standard deep learning objective function [Kautz, 2022; Xu *et al.*, 2018; Marra *et al.*, 2019]. The trained system is a standard NN model that does not utilize rules at test time. In contrast, reasoning approaches typically perform symbolic reasoning [Raedt *et al.*, 2020; Qu *et al.*, 2021] based on rules.

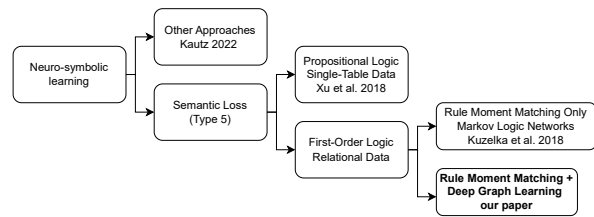


Figure 2: Within neuro-symbolic AI, we develop a new first-order semantic loss approach.

Compared to previous semantic loss approaches [Xu *et al.*, 2018], our main novelty is that *we incorporate knowledge expressed in first-order logic, rather than the less powerful formalism of propositional logic*. For example, a propositional rule would be “if a movie is a horror movie, it is not likely to be a romance”. A first-order rule could be “if a user rates a horror movie, the user is most likely to be a man”. Since first-order rules incorporate relationships, but propositional rules do not, *first-order logic can leverage the full power of relational data*. Previous FOL approaches (e.g., Marra *et al.* [2019]), are based on fuzzy logic, and the satisfaction function is used as a training objective on its own, rather than as a regularizer for a standard graph generative likelihood.

*Deep graph generative models.* The closest predecessor to our work is the constrained VGAE model of Ma *et al.* [2018] where a VGAE likelihood is maximized subject to a constraint of the form  $g(\theta) = 0$ . While this general form covers rule matching, the work of Ma *et al.* does not incorporate first-order logic for specifying graph patterns, nor does it address computing pattern counts.

In principle the rule-enhanced likelihood objective can be used for maximum likelihood training with any deep graph generative model. We selected VGAEs as our base model for several reasons. (1) They are a well-established and widely used GGM. Mahmoudzadeh *et al.* [2024] show that their VGAE+ model is a strong multitask model that provides accurate predictions for a wide range of knowledge graph queries, based on inference from a single model. (2) They support learning from a single large graph, rather than from a set of graphs [Faez *et al.*, 2021]. Rule learners also utilize the single-graph setting [Qian and Schulte, 2015; Meilicke *et al.*, 2024], so the VGAE input data are compatible with the rule learner input data. (3) As we show in this paper, the conditional link independence assumptions of VGAEs facilitates the computation of expected rule instance counts. We believe that extending rule moment matching to other generative models is a fruitful topic for future research.

*Maximum Entropy Moment Matching.* Kuzelka *et al.* [2018] show that a distribution  $P$  over graphs maximizes entropy subject to rule moment matching if and only if  $P$  is defined by a log-linear model known as a *Markov Logic Network* (MLN) with maximum likelihood weights  $w$ . Both the maximum entropy objective and our constrained likelihood objective capture the global graph statistics represented by rule counts. However, the VGAE likelihood can in addition capture local graph patterns. For example, matching the number of observed triangles in a graph is unlikely to capture

community structure, or which nodes have special properties such as centrality.

### 3 Background on First-Order Logic

We review background on logic and rule learning.

#### 3.1 Attributed Heterogenous Graphs

An attributed graph is a pair  $\mathcal{G} = (V, E)$  where  $V$  is a set of nodes of size  $|V| = n$  and  $E \subseteq V \times V$  is a set of edges. Node features are summarized in  $n \times f$  matrix  $\mathbf{X}$  and node labels in a  $n \times L$  matrix  $\mathbf{L}$  where the  $u$ -th row of  $\mathbf{L}$  is a one-hot encoding of the label of node  $u$ . Different edge types are represented by a set of adjacency matrices  $\mathbf{A} = \{\mathbf{A}_1, \dots, \mathbf{A}_T\}$ . The notation  $\mathbf{A}_r[u, v] = 1$  indicates that there is a link  $u \rightarrow_r v$  of type  $r$  from node  $u$  to node  $v$ . Figure 3 shows part of the information in an attributed graph using the tabular SQL format.

User			Rating			Movie			
User Id	Age	Gender	User Id	Movie Id	Rating	Movie Id	Action	Drama	Horror
3	0 (0.34)	M (0.55)	3	The Dictator	1 (0.75)	The Dictator	0 (0.38)	0 (0.73)	0 (0.85)
5	1 (0.43)	F (0.34)	5	Thor	4 (0.36)	Thor	1 (0.49)	0 (0.66)	0 (0.4)
7	2 (0.90)	M (0.84)	5	The Dictator	3 (0.84)	BraveHeart	1 (0.91)	1 (0.41)	1 (0.7)
...	...	...	7	BraveHeart	5 (0.98)	...	...	...	...

Figure 3: Excerpt from a relational dataset. (a) An attributed graph represented in table format. (b) The probabilities assigned to each data entry specify a probabilistic graph (see below).

#### 3.2 First-Order Logic

We follow previous work in SRL [Schulte and Gholami, 2017; Kimmig *et al.*, 2014]. A **population** is a set of individuals of the same type (e.g., a set of *Users*, a set of *Movies*). Individuals are denoted by constants (e.g.,  $user_3$  and  $thor$ ). An attributed graph specifies a set of individuals (nodes) for each type. A **node variable** ranges over a population, and is denoted in upper case such as  $User, Movie, U, V$ . A unary **functor** maps an individual to a value, and corresponds to a node attribute/label. A binary functor maps an ordered pair of individuals to a value, and corresponds to an edge/edge type. Functors are denoted  $f, f'$  etc.

A **first-order** (FOT) term is of the form  $f(U)$  where each population variable  $U_i$  is of the appropriate type. FOT examples are  $age(User)$  and  $rating(User, Movie)$ . A FOT can be instantiated with individual constants, much like an index in a plate model [Kimmig *et al.*, 2014]. A **grounding**  $U = \mathbf{u}$  for a list of FOTs simultaneously replaces each population variable in the list by a constant. (We assume that different population variables are replaced by different constants.) Ground term examples are  $age(User = user_5)$ , and  $rating(User = user_5, Movie = thor)$ . A ground term, in Python-style, assigns individuals as argument to node variables, then applies the functor to return a value.

An FO **literal** is of the form  $\ell \equiv f(U) = v$ . A **conjunction** is a list of literals  $\phi = \ell_1, \dots, \ell_s$ . We write  $\phi(U)$  for an FO conjunction and  $\phi(U = \mathbf{u})$  for a ground conjunction. A graph  $\mathcal{G}$  **satisfies** a ground literal if the graph assigns value  $v$  to the ground term  $f(U = \mathbf{u})$ , and satisfies the conjunction  $\phi$  if it satisfies each ground literal in the conjunction. The

Conjunction $\phi$	$n_\phi(\mathcal{G})$
$Age(User) = 0$	376
$Rating(User, Movie) = 1$	4701
$Age(User) = 0, Rating(User, Movie) = 1$	2524

Table 1: Conjunction Instance counts in the MovieLens database  $\mathcal{G}$

**instance count**  $n_\phi(\mathcal{G})$  in a graph  $\mathcal{G}$  returns the number of  $\phi$ -groundings satisfied by graph  $\mathcal{G}$ .

A **probabilistic graph**  $\tilde{\mathcal{G}}$  assigns a probability  $p_{\tilde{\mathcal{G}}}(\ell(U = \mathbf{u}))$  to each ground literal. The **probabilistic instance count** of a conjunction [Kuzelka, 2023] is the probability product, summed over all conjunction groundings:

$$n_\phi(\tilde{\mathcal{G}}) = \sum_{U=\mathbf{u}} \prod_{i=1}^s p_{\tilde{\mathcal{G}}}(\ell_i(U = \mathbf{u})) \text{ for } \phi = \ell_1, \dots, \ell_s \quad (1)$$

**Examples.**  $Age(User) = 1, rating(User, Movie) = 4$  is an FO conjunction. Its grounding  $age(User = user_5) = 1, rating(User = user_5, Movie = thor) = 4$  is satisfied by the data of Figure 3(a). In the probabilistic graph Figure 3(b), the probability of this conjunction is  $0.43 \times 0.36 = 0.1548$ .

Table 1 illustrates FO instance counts using the MovieLens dataset [Qian and Schulte, 2015]. MovieLens contains 376 users at age level 0. The number of user-movie pairs with a rating of 1 is 4701. The number of such pairs with the user at age level 0 is 2524. An FO conjunction specifies a graph motif, and the instance count is the motif count [Ma *et al.*, 2019] (see Figure 8 for illustration).

#### 3.3 First-Order Logic Rules

A rule is of the form  $B \rightarrow H$  where the body  $B$  is an FO conjunction and the head  $H$  is a single FO literal. An example rule is

$$\underbrace{Age(User) = 1}_{\text{body } B} \rightarrow \underbrace{Rating(User, Movie) = 4}_{\text{head } H}.$$

This rule expresses the knowledge that the age level of a user influences their ratings. A question researched in SRL is how to convert a set of rules to a set of graph features/statistics that support *generative* graph modelling. The recommended answer is to convert each rule to a conjunction  $\phi = (B, H)$  [Domingos and Richardson, 2007, 12.5.3], which is known as *moralization*. A basic result in graphical model theory states that moralization produces an undirected model from a directed one that is equivalent in the sense of representing the same joint distribution [Evans, 2022]. Empirical work on relational data has validated moralization for obtaining useful conjunctive features for generative relational models [Kazemi *et al.*, 2014; Khosravi *et al.*, 2012]; see Appendix A.2 for illustration and details.

### 4 Rule-Enhanced Graph Generation

This section considers how to enhance training a parametrized graph generative model (GGM)  $P_\theta$  on a training graph  $D$ , with a list of conjunctions  $\phi_1, \dots, \phi_k$ .

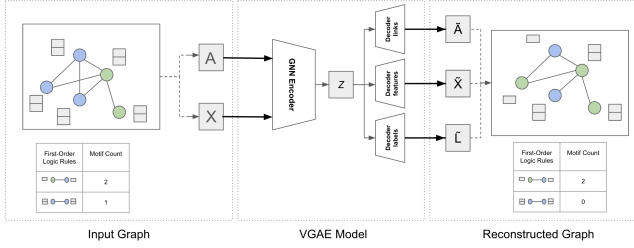


Figure 4: Encoder-Decoder Training Architecture for the VGAE+ Model. Full details are in Appendix A.4.

Our *semantic loss objective* maximizes the data likelihood  $P_\theta(D)$ , subject to the rule moment matching constraint that  $E_\theta[n_i] = n_i(D)$ , where  $n_i(D)$  is the **data instance count** of conjunction  $\phi_i$ , and  $E_\theta[n_i] \equiv \sum_{\mathcal{G}} P_\theta(\mathcal{G})n_i(\mathcal{G})$  is the **expected instance count** for the GGM. For a mixture model GGM, we derive the following Lagrangian evidence lower bound (ELBO) as a training objective.

**Proposition 1.** *Suppose that  $P_\theta(\mathcal{G}) = \int p(\mathcal{G}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$  is a mixture model. Then*

$$\ln P_\theta(D) - \lambda/k \sum_{i=1}^k \rho(n_i(D), E_\theta[n_i]) \geq \quad (2)$$

$$E_{\mathbf{z} \sim q_\phi(\mathbf{z}|D)}[\ln P_\theta(D|\mathbf{z})] - KL(q_\phi(\mathbf{z}|D)||p(\mathbf{z})) \quad (3)$$

$$-\lambda/k \sum_{i=1}^k \rho(n_i(D), E_\theta[n_i|\mathbf{z}]), \quad (4)$$

where  $\rho(\text{count}_1, \text{count}_2) \geq 0$  is a differentiable distance metric that maps two counts to a non-negative number.

Proposition 1 says that the constrained likelihood (2) can be approximated by our new **rule-matching variational ELBO objective** [(3)-(4)]. Equation (2) is the standard ELBO for a graph generative mixture model [Kipf and Welling, 2016]. Equation (4) is the rule-matching semantic loss component. Since an FO conjunction specifies a graph motif, we refer to Equation (4) as a **motif loss** for brevity.

#### 4.1 Implementing the Rule-Matching ELBO

Our novel VGAE+R architecture extends the recent VGAE+ architecture [Mahmoudzadeh *et al.*, 2024] to match rules.

**Encoder-Decoder Architecture.** Figure 4 shows the VGAE+R architecture. The **encoder** model  $q_\phi(\mathbf{z}|D)$  can be any GNN that maps a heterogeneous graph to node embeddings, such as RGCN. The VGAE+R **decoder** independently maps node embeddings to different graph components with three different decoders [Mahmoudzadeh *et al.*, 2024]:

$$\ln P_\theta(D|\mathbf{z}) = [\alpha \ln p_\eta(\mathbf{A}|\mathbf{z}) + \beta \ln p_\psi(\mathbf{X}|\mathbf{z}) + \gamma \ln p_\phi(\mathbf{L}|\mathbf{z})]$$

where  $p_\eta : \mathbb{R}^d \times \mathbb{R}^d \rightarrow [0, 1]$  is a trainable **link decoder**,  $p_\psi$  is a trainable **feature decoder**, and  $p_\phi$  is a trainable **label decoder** (see Figure 4). The hyperparameters  $\alpha, \beta$  and  $\gamma$  weight the importance of different reconstruction tasks.

For the *distance metric*  $\rho$  to compare an expected count to an observed count we use

$$\rho(n_i(D), E_\theta[n_i|\mathbf{z}]) = |\ln n_i(D) - \ln E_\theta[n_i|\mathbf{z}]|.$$

The magnitude of conjunction counts grows exponentially with the number of node variables in the conjunction. Comparing expected counts on a log-scale decreases the impact of the number of variables and improves numeric stability.

**Computing Expected Instance Counts.** The next proposition shows that the expected conjunction count over all graphs can be found as a *probabilistic instance count applied to a single graph*, the expected graph. Given a set of node embeddings  $\mathbf{z}$ , the **expected graph**  $\tilde{\mathcal{G}}_{\mathbf{z}}$  is a probabilistic graph that assigns a probability to each ground literal by applying the decoder to the relevant links/node features/edge types. For examples see Figure 3 and Figure 12.

**Proposition 2.** *The expected conjunction count given a set of node embeddings can be computed as the conjunction count in the expected graph:  $E_\theta[n_i|\mathbf{z}] = n_i(\tilde{\mathcal{G}}_{\mathbf{z}})$ .*

The proof is in the supplement. The upshot is that rule moment matching can be implemented by performing instance counting in a single graph, which we address next.

## 5 Matrix Multiplication for Instance Counting

This section presents a novel matrix multiplication method for instance counting, that is differentiable and applies to both discrete and probabilistic graphs.

To illustrate the basic idea, consider the instance count of the conjunction  $R(U_1, V_1), R(V_1, V_2), R(V_2, U_1)$ , which finds the number of triangles in an undirected graph represented by an adjacency matrix  $\mathbf{A}$ . It is well-known that the triangle count is given by  $\sum_{u=1}^n \mathbf{A}_{u,u}^3$ , the trace of the third power of the adjacency matrix. We generalize this approach to a large class of logical formulas: given a chain conjunction, our algorithm computes a sequence of matrix multiplications such that the conjunction’s instance count can be found by executing the matrix multiplications. For simplicity, we specify our algorithm for binary literals, then discuss extensions.

A **chain conjunction** of binary literals is of the form  $\phi = \ell_1(U_1, V_1), \dots, \ell_P(U_P, V_P)$  where  $V_i = U_{i+1}$  for every  $i$ . Algorithm 1 maps each chain conjunction to a sequence of adjacency matrix multiplications.

**Example.** Consider the chain conjunction

$$\begin{aligned} & \text{AdvisedBy}(\text{Student}, \text{Professor}), \\ & \text{Teaches}(\text{Professor}, \text{Course}), \\ & \text{TakesCourse}(\text{Course}, \text{Student}) \end{aligned}$$

For the example graph of Table 2, Figure 5 shows the corresponding sequence of matrix multiplications.

**Extensions.** Unary literals can be included by omitting nodes from the input graph that do not satisfy them. Probabilistic instance counts can be obtained by using soft matrices  $\tilde{\mathbf{A}}, \tilde{\mathbf{X}}, \tilde{\mathbf{L}}$ ; see Appendix A.7 for details.

**Correctness.** The next proposition shows that the instance count for the chain conjunction can be obtained through summing over the entries in the constructed matrix product.

---

**Algorithm 1** Matrix Multiplication for Instance Counting
 

---

```

1: Input: Chain conjunction  $\phi = \{l_1(U_1, V_1), \dots, l_P(U_P, V_P)\}$ 
2: Output: Instance count  $n_\phi(\mathcal{G})$  or expected count  $n_\phi(\tilde{\mathcal{G}}_z)$ 
3: {Initialize adjacency matrices  $\mathbf{A}_{\ell_k}$  for binary literals  $\ell_k, k = 1, \dots, P$ }
4: for  $k = 1$  to  $P$  do
5:   if positive literal  $\ell_k = R(U_k, V_k) = 1$  then
6:      $\mathbf{A}_{\ell_k} \leftarrow \mathbf{A}_r$ 
7:   else if  $\ell_k = R(U_k, V_k) = 0$  then
8:      $\mathbf{A}_{\ell_k} \leftarrow \neg\mathbf{A}_r$  where  $\neg\mathbf{A}_r$  is the complement of  $\mathbf{A}_r$ 
9:   end if
10: end for
11:  $O_1 \leftarrow \mathbf{A}_{\ell_1}$ 
12: for  $k = 1$  to  $P - 1$  do
13:    $O_{k+1} \leftarrow O_k \cdot \mathbf{A}_{\ell_{k+1}}$ 
14:   if  $V_{k+1} = U_1$  then
15:     Zero out the non-diagonal entries of  $O_{k+1}$ 
16:   end if
17: end for
18: Return:
19:  $n_\phi(\mathcal{G}) = \sum(O_P(\phi))$  for input graph  $\mathcal{G}$ 
20:  $n_\phi(\tilde{\mathcal{G}}_z) = \sum(O_P(\phi))$  for expected graph  $\tilde{\mathcal{G}}_z$ 

```

---

Advised by	
Student	Professor
Jack	Jane
Joey	Tom

Teaches	
Professor	Course
Jane	DL
Tom	DL
Tom	NLP

Takes Course	
Course	Student
DL	Jack
DL	Joey
DL	Lucas
DL	Ellie
NLP	Lucas
NLP	Ellie

Table 2: An example graph represented as tables containing unary and binary relations.

**Proposition 3.** Let  $\phi$  be a centered chain conjunction of length  $k$ , i.e., the first node variable is the only one that appears twice non-consecutively.

- For an input graph  $\mathcal{G}$ , the  $(u, v)$ -th entry of  $O_k$  counts the number of groundings of  $\phi$  in  $\mathcal{G}$  where  $U_1 = u$  and  $V_P = v$ . Therefore  $n_\phi(\mathcal{G}) = \sum(O_k(\phi))$ .
- For an expected graph  $\tilde{\mathcal{G}}_z$ , the  $(u, v)$ -th entry of  $O_k$  counts the expected number of groundings of  $\phi$  where  $U_1 = u$  and  $V_P = v$ . Therefore  $n_\phi(\tilde{\mathcal{G}}_z) = \sum(O_k(\phi))$ .

In our experiments, we found that all learned rules were centered. The Appendix extends matrix multiplication to non-centered chains.

## 6 Evaluation

We detail our methodology and then discuss our empirical results. We will make our code available on-line.

### 6.1 Experimental Design.

We describe our benchmark datasets, comparison methods, and how evaluation metrics are computed.

AdvisedBy(Student, Professor), Teaches(Professor, Course), TakeCourse(Course, Student)

$$\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 2 & 1 & 2 & 1 \end{bmatrix}$$

Figure 5: The matrix multiplication sequence for our example conjunction and table 2 graph. The final result is 2, which is the number of satisfying groundings in the input graph.

### Datasets

We use datasets from previous studies of GGMs [Mahmoudzadeh *et al.*, 2024; Yun *et al.*, 2019; Hao *et al.*, 2020]. Cora, ACM, and CiteSeer are citation networks, IMDb is a movie dataset, and UW represents an academic department. Appendix A.1 presents dataset and preprocessing details.

### Evaluation Metrics

We compare rule-enhanced VGAE+R training with plain VGAE+ training, using three main metrics. In the following, we refer to a complete dataset as the **input graph**. Our evaluation measures *graph realism*—the quality of generated graphs—and the downstream task of node classification.

**Count Distance** Given the training graph  $D$ , we sample one node embedding matrix  $z$  from the encoder posterior  $q_\phi(z|D)$  and then apply the decoder model eq. (5) to  $z$  to obtain the expected graph  $\tilde{D}_z$ . We report the mean squared distance  $(1/k \sum_{i=1}^k [n_i(D) - n_i(\tilde{D}_z)]^2)^{1/2}$ —between the observed motif counts and the expected motif counts in the reconstructed graph—as the **count distance** (CD), where  $k$  is the number of rules.

**Graph Realism** measures how similar graphs generated by the model are to observed graphs. How to quantitatively assess generated graphs has been studied in recent papers. We adapt the SOTA approach that compares graph embeddings of the training graph to embeddings of generated graphs using Maximum Mean Distance (MMD) [O’Bray *et al.*, 2022; Thompson *et al.*, 2022; Shirzad *et al.*, 2022]; see Appendix A.8 for details. Our measurements utilize the MMD metric with a linear kernel, as recommended by Thompson *et al.* [2022]. Unlike Count Distance, the MMD metric is independent of the training objective.

**Node Classification** To compute a node classification score, we randomly divide the nodes in the input graph into training and test nodes (80%/20%). The training graph is the input graph but with the test node labels removed. At test time, we run the encoder on the input graph to obtain node embeddings for all nodes, then apply the decoder to predict node labels for the test nodes.

### 6.2 Experimental Results

Count Distance and Graph Realism are the most important metrics for us since they directly pertain to graph generation quality. Our graph generation baseline is the VGAE+ model trained without rule matching (i.e.,  $\lambda = 0$ ). To obtain rules

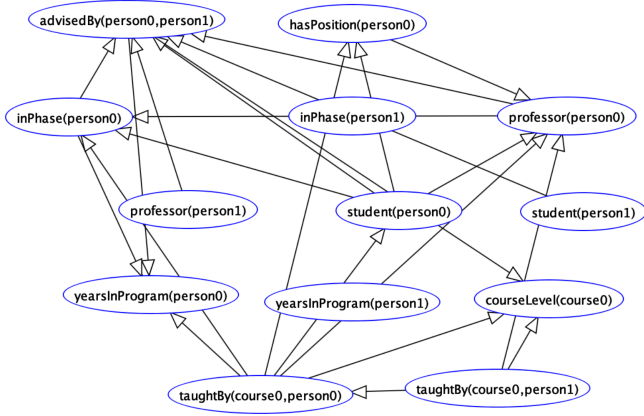


Figure 6: The BN graph for the UW dataset used to train the VGAE+R model.

Table 3: Count Distance (MSE $\downarrow$ ) and Improvements in %.

Dataset	VGAE+	VGAE+R	Improvement (%)
<b>Cora</b>	112018.5	<b>36608.7</b>	67.33
<b>CiteSeer</b>	47963.6	<b>20614.9</b>	57.00
<b>IMDb</b>	981433.1	<b>108584.2</b>	88.94
<b>ACM</b>	101733.1	<b>34396.8</b>	66.20
<b>Computers</b>	7809684.3	<b>229343.2</b>	97.06
<b>Photo</b>	94124.1	<b>52918.8</b>	43.76
<b>UW</b>	616966.8	<b>545615.0</b>	11.56

for training a VGAE+R model, we used the Factorbase system [Qian and Schulte, 2015] with default settings. Factorbase outputs a FOBN whose structure defines a set of rules  $B \rightarrow H$ . See Appendix A.3 for details on FO rule learning. Figure 6 shows the BN graph for the UW dataset. It captures several patterns that express university domain knowledge, such as the following. (1) Whether a person teaches a course correlates with whether they have a position. (2) Course teachers are more likely to be professors. (3) A person’s program phase predicts their years in the program.

### Count Distance and Graph Realism

Table 3 shows the difference between expected and observed rule instance counts. Both methods show large absolute distances because a VGAE model tends to produce overly dense graphs [Orbanz and Roy, 2014]. However in comparison, we observe a *very large improvement in the match between expected and observed counts*, at least 36% on all datasets, except for the small graph UW with an improvement of 11.56%. This shows that VGAE training without the moment matching constraint is far from matching rule counts.

On the graph realism metric shown in Table 4, we again find large absolute distances with the training set, and *very large improvements through rule learning, by an order of magnitude*. Overall we conclude that unconstrained VGAE training does not match the instantiation counts of the learned rules, and that moment matching has a large impact on the realism of the graphs generated.

Table 4: Graph Realism (MMD $\downarrow$ ) with Log Improvement.

Dataset	VGAE+	VGAE+R	Log Improv.
<b>Cora</b>	$5.0 \times 10^{18}$	$2.4 \times 10^{17}$	1.32
<b>CiteSeer</b>	$1.5 \times 10^{18}$	$9.1 \times 10^{16}$	1.23
<b>IMDb</b>	$1.7 \times 10^{23}$	$1.9 \times 10^{18}$	5.89
<b>ACM</b>	$2.7 \times 10^{20}$	$7.3 \times 10^{18}$	1.57
<b>Computers</b>	$1.4 \times 10^{25}$	$2.1 \times 10^{24}$	0.84
<b>Photo</b>	$5.1 \times 10^{23}$	$7.4 \times 10^{21}$	1.85
<b>UW</b>	$8.8 \times 10^{10}$	$6.4 \times 10^9$	1.13

### Node Classification

Node classification is an extensively studied task in graph learning. Since SOTA performance is nearly saturated, we do not claim that VGAE+R leads to uniformly best node classification. Instead we investigate two hypotheses:

1. Rule enhancement can improve GGM-based classification when the rules capture relevant domain knowledge.
2. The VGAE+R model is competitive with current baselines.

The baselines apply to homogeneous graphs only, so we homogenized ACM and IMDb [Mahmoudzadeh *et al.*, 2024] to create ACM\* and IMDB\* and omitted UW. Table 5 shows an improvement on 4 out of 6 datasets for rule enhancement, substantive for two of them (Cora and UW). The biggest improvement is on Cora, where the rules increase the AUC score by 10%. Rule enhancement decreases classification performance only slightly even when the rules are not very relevant for the class label.

Dataset	Model	AUC	F1 Score
Cora	VGAE+R	<b>0.965 <math>\pm</math> 0.013</b>	<b>0.887 <math>\pm</math> 0.016</b>
	VGAE+	0.865 $\pm$ 0.043	0.699 $\pm$ 0.103
UW	VGAE+R	<b>0.794 <math>\pm</math> 0.000</b>	<b>0.6323 <math>\pm</math> 0.008</b>
	VGAE+	0.683 $\pm$ 0.000	0.6154 $\pm$ 0.008
CiteSeer	VGAE+R	<b>0.903 <math>\pm</math> 0.008</b>	<b>0.794 <math>\pm</math> 0.042</b>
	VGAE+	0.891 $\pm$ 0.013	0.733 $\pm$ 0.058
Computers	VGAE+R	0.915 $\pm$ 0.022	0.827 $\pm$ 0.047
	VGAE+	<b>0.920 <math>\pm</math> 0.004</b>	<b>0.837 <math>\pm</math> 0.005</b>
Photo	VGAE+R	<b>0.991 <math>\pm</math> 0.003</b>	<b>0.972 <math>\pm</math> 0.002</b>
	VGAE+	0.980 $\pm$ 0.021	0.946 $\pm$ 0.052
ACM*	VGAE+R	0.761 $\pm$ 0.077	<b>0.525 <math>\pm</math> 0.014</b>
	VGAE+	<b>0.775 <math>\pm</math> 0.074</b>	0.523 $\pm$ 0.009
IMDb*	VGAE+R	<b>0.829 <math>\pm</math> 0.006</b>	<b>0.697 <math>\pm</math> 0.008</b>
	VGAE+	0.828 $\pm$ 0.011	0.687 $\pm$ 0.014

Table 5: Node classification results for graph generation with and without rule enhancement.

Table 6 compares the rule-enhanced VGAE+R with recent node classification baselines (see Appendix A.9). We focused on baselines that are similar to generative models in that they aim to support multiple prediction tasks. The baselines apply to homogeneous graphs only, so we homogenized ACM and IMDb Mahmoudzadeh *et al.* [2024] and omitted UW.

Dataset	Model	AUC	F1-score
Cora	VGAE+R	<b>0.965 ± 0.013</b>	<b>0.887 ± 0.016</b>
	GiGaMAE	0.920	0.856
	$G^2P_{xy}$	0.921	0.781
	MVGRL	0.888	0.886
CiteSeer	VGAE+R	<b>0.903 ± 0.008</b>	0.794 ± 0.042
	GiGaMAE	0.842	<b>0.798</b>
	$G^2P_{xy}$	0.850	0.781
	MVGRL	0.807	0.710
Computers	VGAE+R	0.915 ± 0.022	<b>0.827 ± 0.047</b>
	GiGaMAE	0.941	0.770
	$G^2P_{xy}$	0.680	0.578
	MVGRL	<b>0.983</b>	0.816
Photo	VGAE+R	<b>0.991 ± 0.003</b>	<b>0.972 ± 0.002</b>
	GiGaMAE	0.963	0.569
	$G^2P_{xy}$	0.773	0.513
	MVGRL	0.963	0.960
ACM*	VGAE+R	0.761 ± 0.084	0.525 ± 0.014
	GiGaMAE	<b>0.823</b>	0.440
	$G^2P_{xy}$	0.742	0.661
	MVGRL	0.708	<b>0.803</b>
IMDb*	VGAE+R	0.829 ± 0.006	<b>0.697 ± 0.008</b>
	GiGaMAE	<b>0.890</b>	0.457
	$G^2P_{xy}$	0.654	0.541
	MVGRL	0.788	0.653

Table 6: Node classification results comparing rule-enhanced graph generation against baselines.

In Table 6, our VGAE+R model shows the best node classification performance on 3 out of 6 datasets. The biggest improvement is on CiteSeer where our baselines are far from SOTA performance. GiGaMAE is a strong baseline that achieves the best result on two datasets. Our conclusion is that *rule-enhanced graph generation supports node classification that is competitive with recent baselines*.

**Learning Curve** After learning an informative set of rules on the entire training graph, we sample 20% of node labels as test labels and reserve the other 80% as training node labels. Then we sample  $x\%$  of the training node labels for training the VGAE with and without rules. We report the predictive accuracy on the test labels, for  $x = 25\%, 50\%, 75\%, 100\%$  of training labels. The idea is to simulate the impact of a domain expert providing the model with a strong set of rules. Figure 7 shows that *rule matching improves data efficiency substantially on the CiteSeer dataset*, where rule learning achieves a higher accuracy with 50% of node labels than baseline VGAE learning with 100%. We see similar efficiency improvements on Cora and Photo. The learning curves with and without rules are similar for the datasets Computers, ACM, and IMDb because the rules do not affect node classification as much as they affect graph generation. UW is too small to obtain a meaningful learning curve.

## 7 Conclusion and Future Work

We proposed a new semantic loss objective function for training a deep graph generative model (GGM) to incorporate

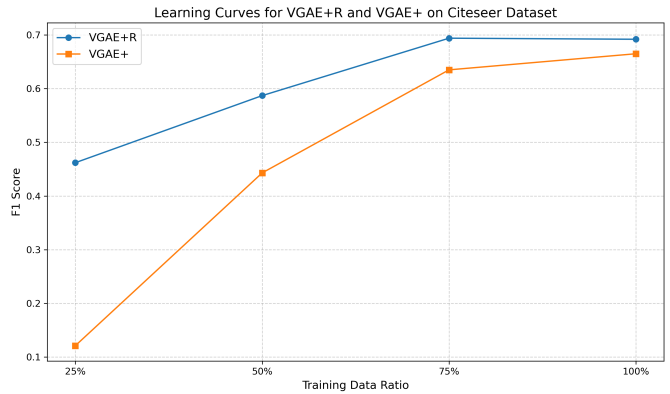


Figure 7: Learning curve for the CiteSeer dataset.

domain knowledge expressed by logical rules: Maximize the data likelihood subject to a moment matching constraint, which requires the expected rule instance counts under a model to match the observed rule instance count. Our main algorithmic contribution is a new differentiable matrix multiplication method for computing both observed and expected counts. In empirical evaluation, rule matching improves the quality of the graphs generated by a Variational Graph Auto-Encoder (VGAE) model by an order of magnitude or more, both with respect to rule counts and with respect to a standard metric of graph realism. Applying the trained GGM to the downstream task of node classification, rule matching improved classification accuracy on all but one of our benchmark datasets. The domain knowledge incorporated in the model is often effective in improving predictions from small datasets, as shown in learning curves.

The main limitation of our current approach is scaling the matrix multiplication algorithm for expected counts. A possible approach would be approximation algorithms from the related problem of weighted model counting [van Bremen and Kuzelka, 2020].

Our novel approach to rule-enhanced graph learning opens several avenues for future research, including (1) Extracting different rule statistics for moment matching (e.g. conditional probabilities from if-then rules). (2) Evaluating methods other than moment-matching for measuring how well a set of FO rules fits a graph (e.g., Marra *et al.* [2019]; Badreddine *et al.* [2022]). (3) Enhancing different types of GGMs with rule matching (e.g., diffusion and auto-regressive models).

In sum, rule moment matching presents a novel semantic loss approach to neuro-symbolic AI that combines logical rules with deep graph learning. Our experiments show great potential for enhancing deep graph generative models with rule-based knowledge.

## References

- Samy Badreddine, Artur d’Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artificial Intelligence*, 303:103649, 2022.
- Pedro Domingos and Daniel Lowd. Unifying logical and

- statistical AI with Markov logic. *Communications of the ACM*, 62(7):74–83, 2019.
- Pedro Domingos and Matthew Richardson. Markov logic: A unifying framework for statistical relational learning. In *Introduction to Statistical Relational Learning*. MIT Press, 2007.
- Robin J. Evans. *Graphical Models: DAGs, UGs, and Beyond*. Cambridge University Press, Cambridge, 2022.
- Faezeh Faez, Yassaman Ommi, Mahdieh Soleymani Baghshah, and Hamid R Rabiee. Deep graph generators: A survey. *IEEE Access*, 9:106675–106702, 2021.
- Artur d’Avila Garcez and Luis C Lamb. Neurosymbolic AI: The 3rd wave. *Artificial Intelligence Review*, pages 1–20, 2023.
- William L Hamilton. *Graph representation learning*, volume 14. Morgan & Claypool Publishers, 2020.
- Yu Hao, Xin Cao, Yixiang Fang, Xike Xie, and Sibao Wang. Inductive link prediction for nodes having only attribute information. *arXiv preprint arXiv:2007.08053*, 2020.
- Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International conference on machine learning*, pages 4116–4126. PMLR, 2020.
- Manfred Jaeger and Oliver Schulte. A complete characterization of projectivity for statistical relational models. In *International Joint Conferences on Artificial Intelligence IJCAI-20*, pages 4283–4290, 2020.
- Henry Kautz. The third AI summer: AAI Robert S. Englemore Memorial Lecture. *AI magazine*, 43(1):105–125, 2022.
- Seyed Mehran Kazemi, David Buchman, Kristian Kersting, Sriraam Natarajan, and David Poole. Relational logistic regression. In *International Conference on the Principles of Knowledge Representation and Reasoning, KRR 2014*, 2014.
- Hassan Khosravi, Oliver Schulte, Jianfeng Hu, and Tianxing Gao. Learning compact Markov logic networks with decision trees. *Machine Learning*, 89(3):257–277, 2012.
- Angelika Kimmig, Lilyana Mihalkova, and Lise Getoor. Lifted graphical models: a survey. *Machine Learning*, 99(1):1–45, 2014.
- Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.
- Ondrej Kuzelka, Yuyi Wang, Jesse Davis, and Steven Schockaert. Relational marginal problems: Theory and estimation. In *AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 1–8, 2018.
- Ondrej Kuzelka. Counting and sampling models in first-order logic. In *International Joint Conference on Artificial Intelligence, IJCAI*, pages 7020–7025, 2023.
- Tengfei Ma, Jie Chen, and Cao Xiao. Constrained generation of semantically valid graphs via regularizing variational autoencoders. In *Advances in Neural Information Processing Systems, NeurIPS 2018*, pages 7113–7124, 2018.
- Chenhao Ma, Reynold Cheng, Laks VS Lakshmanan, Tobias Grubenmann, Yixiang Fang, and Xiaodong Li. Linc: a motif counting algorithm for uncertain graphs. *VLDB Endowment*, 13(2):155–168, 2019.
- Erfaneh Mahmoudzadeh, Parnis Naddaf, Kiarash Zahirnia, and Oliver Schulte. Deep generative models for subgraph prediction. In *European Conference on Artificial Intelligence, ECAI 2024*, volume 392, pages 3128–3136, 2024.
- Giuseppe Marra, Francesco Giannini, Michelangelo Dili-genti, and Marco Gori. Lyrics: A general interface layer to integrate logic inference and deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD*, pages 283–298, 2019.
- Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. *ACM*, 2015.
- Christian Meilicke, Melisachew Wudage Chekol, Patrick Betz, Manuel Fink, and Heiner Stuckeschmidt. Anytime bottom-up rule learning for large-scale knowledge graph completion. *The VLDB Journal*, 33(1):131–161, 2024.
- Leslie O’Bray, Max Horn, Bastian Rieck, and Karsten Borgwardt. Evaluation metrics for graph generative models: Problems, pitfalls, and practical solutions. In *International Conference on Learning Representations, ICLR*, 2022.
- Peter Orbanz and Daniel M Roy. Bayesian models of graphs, arrays and other exchangeable random structures. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):437–461, 2014.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
- J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge university press, 2000.
- Zhensong Qian and Oliver Schulte. Factorbase: Multi-relational model learning with sql all the way. In *Data Science and Advanced Analytics (DSAA)*, pages 1–10, 2015.
- Meng Qu, Junkun Chen, Louis-Pascal Xhonneux, Yoshua Bengio, and Jian Tang. RNNLogic: Learning logic rules for reasoning on knowledge graphs. In *International Conference on Learning Representations, ICLR*, 2021.
- Luc De Raedt, Kristian Kersting, Sriraam Natarajan, and David Poole. Statistical relational artificial intelligence: Logic, probability, and computation. *Synthesis lectures on artificial intelligence and machine learning*, 10(2):1–189, 2016.
- Luc de Raedt, Sebastijan Dumančić, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In *International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4943–4950, 2020.
- Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.
- Stuart Russell. Unifying logic and probability. *Communications of the ACM*, 58(7):88–97, 2015.

- Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *International conference on the semantic web (ESWC 2018)*, pages 593–607, 2018.
- Oliver Schulte and Sajjad Gholami. Locally consistent Bayesian network scores for multi-relational data. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2693–2700, 2017.
- Oliver Schulte and Zhensong Qian. FACTORBASE: multi-relational structure learning with SQL all the way. *International Journal of Data Science and Analytics*, 7(4):1–21, 2018.
- Yucheng Shi, Yushun Dong, Qiaoyu Tan, Jundong Li, and Ninghao Liu. Gigamae: Generalizable graph masked autoencoder via collaborative latent space reconstruction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 2259–2269, 2023.
- Hamed Shirzad, Kaveh Hassani, and Danica J Sutherland. Evaluating graph generative models with contrastively learned features. *Advances in Neural Information Processing Systems, NeurIPS*, 2022.
- P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT Press, 2000.
- Rylee Thompson, Boris Knyazev, Elahe Ghalebi, Jungtaek Kim, and Graham W Taylor. On evaluation metrics for graph generative models. In *International Conference on Learning Representations, ICLR*, 2022.
- Yijun Tian, Shichao Pei, Xiangliang Zhang, Wei Wang, Hanghang Tong, and Nitesh V. Chawla, editors. *Knowledge-enhanced Graph Learning*. Workshop at AAAI, 2024.
- Timothy van Bremen and Ondrej Kuzelka. Approximate weighted first-order model counting: Exploiting fast approximate model counters and symmetry. In *International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4252–4258, 7 2020.
- Daisy Zhe Wang, Eirinaios Michelakis, Minos Garofalakis, and Joseph M Hellerstein. Bayesstore: managing large, uncertain data repositories with probabilistic graphical models. In *VLDB*, pages 340–351, 2008.
- Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. In *International Conference on Machine Learning, ICML*, volume 80, pages 5502–5511, 2018.
- Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. *Neural Information Processing Systems, NeurIPS*, 32, 2019.
- Qin Zhang, Zelin Shi, Xiaolin Zhang, Xiaojun Chen, Philippe Fournier-Viger, and Shirui Pan. G2pxy: generative open-set node classification on graphs with proxy unknowns. In *International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 4576–4583, 2023.

## A Technical Appendix

### A.1 Dataset Information

To evaluate all the methods we utilize 5 datasets used in previous studies of generative models Kipf and Welling [2016]; Yun *et al.* [2019]; Hao *et al.* [2020].

- **Cora** [Kipf and Welling, 2016] is a citation dataset that consists of nodes that represent machine-learning papers divided into seven classes (the subjects of the papers) and links that represent citation between them. The target for node classification is the subject of the papers. This dataset has 5,429 links, 2,708 nodes with an average node degree of 3.8.
- **ACM** [Yun *et al.*, 2019] is a citation dataset. It has three types of nodes (paper, author, and venue) and four types of links. The target for node classification is predicting one of three labels corresponding to the conferences where the papers were published. This dataset has 18,929 links, 8,993 nodes with an average node degree of 2.209.
- **IMDb** [Yun *et al.*, 2019] is a movie dataset with three types of nodes (movies, actors, and directors) and it uses the genre of movies as their labels. The target for node classification is predicting one of three genres of movies. This dataset has 19,120 links, 12,772 nodes with an average node degree of 2.9.
- **CiteSeer** [Kipf and Welling, 2016] is also a citation dataset that consists of nodes that represent machine-learning papers divided into six classes (the topics of the papers) and links that represent citation between them. The target for node classification is the topic of the publications. This dataset has 4,732 links, 3,327 nodes with an average node degree of 2.7.
- The **UW** dataset models academic relationships at a university, where persons can be both students and professors and key relations include AdvisedBy (a student advised by a professor) and TaughtBy (a student taking a course taught by a professor). The dataset contains 278 person entities and 132 course entities. The classification target is the phase of study of the student, which can be one of three labels: *pre-quals*, *post-quals*, or *post-generals*. If the person is not a student, the label is 0.
- **Photo & Computers** are datasets from the Amazon co-purchase graph [McAuley *et al.*, 2015]. In these datasets, nodes represent goods, links indicate that two goods are frequently bought together, node features are bag-of-words encoded product reviews, and class labels are given by the product category [Hao *et al.*, 2020].

**Data Preprocessing** Following previous work [Kipf and Welling, 2016], for GNN message passing we add self-loops and make all links undirected (i.e., if the training data contains an adjacency,  $v \rightarrow u$ , it also contains  $u \rightarrow v$ .) Rule learning is applied to the original data.

### A.2 First-order Rules, Conjunctions, and Motifs

Figure 8 illustrates how moralization converts a rule to a conjunction, and how a conjunction represents a motif. In terms

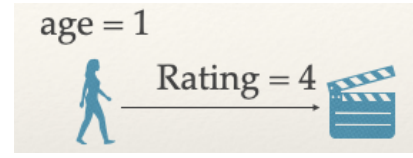


Figure 8: The FO conjunction  $Age(User) = 1, Rating(User, Movie) = 4$  can be viewed as a two-node motif. The conjunction is obtained from the rule  $Age(User) = 1 \rightarrow Rating(User, Movie) = 4$  by moralization.

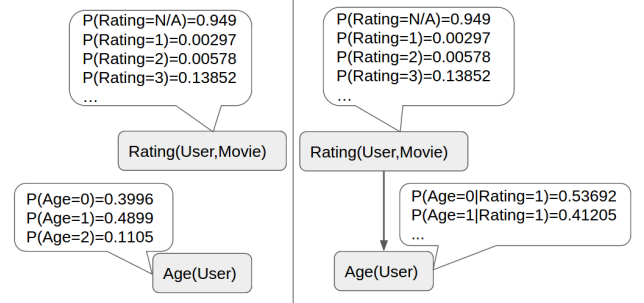


Figure 9: Example First-Order Bayesian Networks: left =  $B_1$  with graph  $\mathcal{G}_1$ , right =  $B_1^+$  with graph  $\mathcal{G}_1^+$ .

of graphical model structure, moralizing converts a Bayesian network to an undirected network (known as a Markov random field), by connecting all parents of a child to each other, and turning all directed links into undirected links. A well-known result in graphical model theory states that if  $P_B(\mathbf{X} = \mathbf{x})$  is the joint distribution over a set of random variables  $\mathbf{X}$  represented by a Bayesian network, and  $M$  is the Markov random field obtained from  $B$  through randomization, then  $M$  can be parametrized to represent  $P_B$ . Further details can be found in texts on graphical models, such as Evans [2022].

### A.3 Rule Learning and First-Order Bayesian Networks

A **Bayesian network (BN) structure** is a directed acyclic graph  $G$  (DAG) whose nodes comprise a set of random variables Pearl [1988]. The causal interpretation is that the parents of a node represent its direct causes Pearl [2000]. A **Bayesian network  $B$**  is a structure  $G$  together with a set of parameter values of the form  $P(child\_value | parent\_values)$ , which specify the distribution of a child node given an assignment of values to its parent node. A first-order BN Wang *et al.* [2008]; Kimmig *et al.* [2014] (FOBN) is a BN whose nodes are first-order terms. The BN parameters specify the distribution of a child node given an assignment of values to its parent node. Figure 9 shows two parametrized FOBNs. The right FOBN connects the rating of a movie to the age of the rater. In the left FOBN, ratings are independent of ages. The process of structure learning searches for statistically significant connections between first-order terms to introduce or remove edges [Schulte and Qian, 2018; Schulte and Gholami, 2017].

A single rule  $B \rightarrow H$  is defined by a combination  $parent\_values \rightarrow child\_value$ , for each child node value and possible combination of parents values [Khosravi *et al.*,

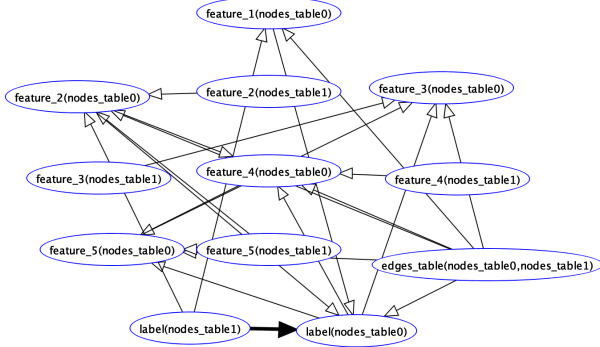


Figure 10: The BN graph for the Cora dataset. We have highlighted the homophily edge that shows that if paper 0 cites paper 1, then the label of paper 0 correlates with the label of paper 1.

2012]. For example, the BN graph of Figure 9(right) defines the rule

$$\text{Rating}(\text{User}, \text{Movie}) = 3 \rightarrow \text{Age}(\text{User}) = 1).$$

Given 6 possible rating values and 5 age levels, the link  $\text{Rating}(\text{User}, \text{Movie}) \rightarrow \text{Age}(\text{User})$  defines  $6 \times 5 = 30$  rules.

Factorbase is the most scalable system for learning a FO generative graphical model with state-of-the-art predictive performance. As the Factorbase system can output many rules, we use

$2n(\mathbf{B}, H) \ln(n(H|\mathbf{B})/n(H)) - \ln n(H)$  as a *rule quality metric* for pruning. The conditional count is given by  $n(H|\mathbf{B}) = n(H, \mathbf{B})/n(\mathbf{B})$ . This metric computes the increase in the log-probability of the head given the body, relative to the prior probability of the head, together with a BIC-type correction for sample size Schulte and Gholami [2017]. The quality metric can be interpreted as a positive local BIC model selection score. We keep all rules whose quality metric is above 0. Rule pruning reduces the number of rules for scalability, and also simulates the impact of a domain expert selecting the most important rules.

We show the BN graphs learned by Factorbase for three datasets. We find that many of the learned rules capture intuitively plausible domain constraints, such as homophily: if one paper cites another, then they are likely from the same research area. For example in the graph for the Cora dataset, the parents of the target label of a paper (node) are paper features 1 and 2, the citation relationship between a paper and the target paper (denoted “edges.table”), and the label of the citing paper. The graph defines a rule for each combination of the 4 parent nodes, and each of the 7 possible child node values (paper classes).

ACM and CiteSeer are citation networks like Cora with a similar graph, which we omit.

Figure 11 shows the BN graph for the IMDb dataset. The graph captures many correlations between features of movies and actors. An interesting aspect of the graph is that for the target label (genre of the movie), its only parents are other features of the movie to be classified (features 1,2,4). This suggests that relational features, such as the features of actors

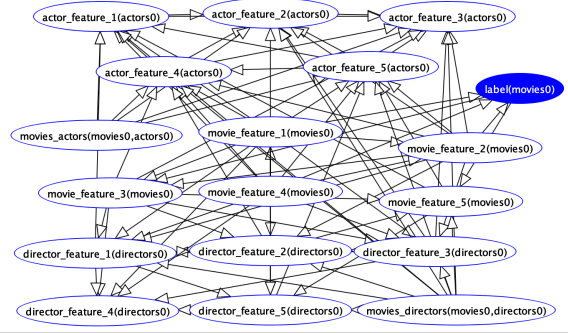


Figure 11: The BN graph for the IMDb dataset.

who appear in the movie, should not add information beyond movie features. We verified this directly by building a classifier based on movie features only, which performed better than the GNN that uses relational features.

#### A.4 Details on the VGAE+R Model

The decoders generate links, node features, and node labels independently as follows.

$$p_{\eta}(\mathbf{A}|\mathbf{z}) = \prod_{r=1}^T \prod_{u,v} p_{\eta_r}(\mathbf{A}_r[u,v]|\mathbf{z}[u], \mathbf{z}[v]) \quad (5)$$

$$p_{\psi}(\mathbf{X}|\mathbf{z}) = \prod_u p_{\psi}(\mathbf{X}[u]|\mathbf{z}[u]) \quad (6)$$

$$p_{\phi}(\mathbf{L}|\mathbf{z}) = \prod_u p_{\phi}(\mathbf{L}[u]|\mathbf{z}[u]), \quad (7)$$

Jaeger and Schulte [2020] provide a strong theoretical foundation for the independent decoder model.

**Hyperparameters** For the encoder, we used an RGCN [Schlichtkrull *et al.*, 2018], with the Pytorch Geometric implementation. The node embedding dimension was 64, the number of RGCN layers 2, with separate layers for estimating the Gaussian posterior mean and the Gaussian posterior standard deviation for each node. For the VGAE+ model with  $\lambda = 0$ , we follow Mahmoudzadeh *et al.* [2024] and use Bayesian optimization to fix the ELBO hyperparameters  $p_{\eta}, p_{\psi}, p_{\phi}$ . For the VGAE+R model, after fixing the ELBO hyperparameters, we set the motif loss weight  $\lambda$  empirically based on a validation set. For the learning curves, we trained the model for 100 epochs at each sample size. For the other graph generative model results, we trained until convergence. The number of training epochs were as follows. 700 for Cora and CiteSeer, 1000 for IMDb, Computers, ACM, Photo, and 100 for UW.

#### A.5 Expanded First-Order Logic Definitions

We add to the definitions in the main body to introduce concepts we need in our proofs and in the full description of matrix multiplication algorithm. We make this section self-contained for ease of reference. A positive **relationship literal** is of the form  $R(U, V)$ . A **negative** relationship literal is of the form  $\neg R(U, V)$ . A generic relationship literal (positive



$$\begin{aligned}
& E_{\mathcal{G} \sim P(\mathcal{G}|\mathbf{z})} [n_\phi(\mathcal{G})] \\
&= E \left[ \sum_{\langle \mathbf{u}, \mathbf{v}, \mathbf{w} \rangle \in \text{Valid}_{EQ}} \prod_{i=1}^P I_{\mathcal{G}}(\ell_i(U_i = u_i, V_i = v_i)) \right. \\
&\quad \left. \prod_{j=1}^Q I_{\mathcal{G}}(\ell_j(W_j = w_j)) \right] \\
&= E \left[ \sum_{\langle \mathbf{u}, \mathbf{v}, \mathbf{w} \rangle \in \text{Valid}_{EQ}} \prod_{i=1}^P l_i^{\mathbf{u}v_i} \prod_{j=1}^Q l_j^{\mathbf{w}^j} \right] \\
&= \sum_{\langle \mathbf{u}, \mathbf{v}, \mathbf{w} \rangle \in \text{Valid}_{EQ}} E \left[ \prod_{i=1}^P l_i^{\mathbf{u}v_i} \prod_{j=1}^Q l_j^{\mathbf{w}^j} \right] \\
&= \sum_{\langle \mathbf{u}, \mathbf{v}, \mathbf{w} \rangle \in \text{Valid}_{EQ}} \prod_{i=1}^P E[l_i^{\mathbf{u}v_i}] \prod_{j=1}^Q E[l_j^{\mathbf{w}^j}] \\
&= \sum_{\langle \mathbf{u}, \mathbf{v}, \mathbf{w} \rangle \in \text{Valid}_{EQ}} \prod_{i=1}^P p_{\tilde{\mathcal{G}}_z}(\ell_i(U_i = \mathbf{u}_i, V_i = \mathbf{v}_i)) \\
&\quad \prod_{j=1}^Q p_{\tilde{\mathcal{G}}_z}(\ell_j(W_j = \mathbf{w}_j)) = n_\phi(\tilde{\mathcal{G}}_z)
\end{aligned} \tag{8}$$

Equation (8) follows because the expectation of a product of independent random variables is the product of their expectations. The random variables  $l_i^{\mathbf{u}v_i}$  and  $l_j^{\mathbf{w}^j}$  are independent because the (in)equality constraints ensure that in a valid grounding, no two different literals are ground to the same ground literal. And conditional on the node embeddings  $\mathbf{z}$ , any two different ground literals are independent.  $\square$

## A.7 Matrix Multiplication Method

### Example for Expected Instance Count

Figure 13 illustrates the matrix multiplication method for the expected graph in our example.



Figure 13: The matrix multiplication sequence for our example conjunction and Figure 12 graph.

### Full Specification of Matrix Multiplication Method

Intuitively, a chain conjunction is a template for a motif or frequently occurring subgraph. The indicator function specifies which nodes satisfy the literal/conjunction. Our formal definitions are as follows.

The input to our counting algorithm is a chain conjunction  $\{\ell_1(U_1, V_1), \dots, \ell_P(U_P, V_P), \ell_1(W_1), \dots, \ell_Q(W_Q), EQ\}$  and an input graph  $\mathcal{G}$ . The first step is to process the unary literals by masking adjacency matrix entries of nodes that do not satisfy all unary literals. The second step is to define inductively a sequence of matrices such that the instance

count of the conjunction can be computed as the entry sum of the matrix product. We use  $A \circ B$  to denote the element-wise matrix (Hadamard) product and  $I$  for the identity matrix of the appropriate dimension. A positive relationship literal  $R(U, V)$  is **associated with**  $\mathbf{A}_r$ , the adjacency matrix for relation  $R$ . A negative relationship literal  $\neg R(U, V)$  is associated with  $\neg \mathbf{A}_r$  where  $\neg \mathbf{A}_r[u, v] = 1 - \mathbf{A}_r[u, v]$  for all node indices  $u, v$ .

**Step 1: Unary Literals** Consider binary relationship literal  $\ell_i(U, V)$  with associated  $m \times n$  adjacency matrix  $A_i(\mathcal{G})$ . We search for every unary literal  $\ell(W)$  where  $(U = W) \in EQ$ . For each such literal, we create a binary vector  $T$  of size  $m$  such that  $T[w] := I_{\mathcal{G}}(\ell(W = w))$ . Thus the entry  $T[w]$  masks all the nodes that do not satisfy the unary literal. We apply the mask to the  $w$  row of matrix  $A_i$ , setting  $\bar{A}_i[w, :] = A_i[w, :] \circ T[w]$ , where  $A(w, \cdot)$  represents the entire  $w$  row of matrix  $A$ . If the  $w$  entry of  $T$  is zero, the entire row of  $\bar{A}_i$  is set to 0. If the  $w$  entry of  $T$  is one, the entire row of  $A_i$  is copied to  $\bar{A}_i$ .

Similarly, if a unary literal  $\ell(W)$  exists where  $(V = W) \in EQ$ , we mask the corresponding column entries in the adjacency matrix  $A_i$ , and repeat the masking process for all such unary literals. We refer to the adjacency matrix that incorporates the unary functor constraints as the *masked adjacency matrix*  $\bar{A}_i$ .

**Step 2: Binary Literals** A chain conjunction is centered if all equality constraints for the binary literals (other than the chain constraints) involve the first node variable, that is they are of the form  $U_1 = E_k$ . For a centered chain, we define a sequence of matrix multiplications as follows.

1. For a single literal conjunct  $\phi = \ell(U, V)$  with associated masked matrix  $\bar{A}$ , let

$$O_1(\phi) = \begin{cases} \bar{A}, & \text{if } U = V \notin EQ \\ \bar{A} \circ I, & \text{if } U = V \in EQ \end{cases}$$

$\bar{A} \circ I$  agrees with  $\bar{A}$  on the diagonal and is 0 off-diagonal.

2. Inductively, consider a conjunction  $\phi$  of length  $k+1$  in the form of  $\phi = \phi', \ell_{k+1}(U_{k+1}, V_{k+1})$  where  $\phi' = \ell_1(U_1, V_1), \dots, \ell_k(U_k, V_k)$  is a conjunction of length  $k$ . Let

$$O_{k+1}(\phi) = \begin{cases} O_k(\phi') \bar{A}_{k+1}, & \text{if } U_1 = V_{k+1} \notin EQ \\ (O_k(\phi') \bar{A}_{k+1}) \circ I, & \text{if } U_1 = V_{k+1} \in EQ \end{cases} \tag{9}$$

### Correctness Proof

We next formulate the proposition that for every chain conjunction, there is a corresponding sequence of matrix multiplication operations such that: for every input graph  $\mathcal{G}$ , applying the operation sequence to the graph edge label tensor returns the instance count. In this formulation, we use these facts: 1) Every ground positive (negative) relationship literal corresponds to a link present (absent) in the graph. 2) A grounded chain conjunction corresponds to a path in the graph where each consecutive pair of nodes is connected by a present/absent link.

**Proposition.** chain conjunction of length  $k$ .

1. For an input graph  $\mathcal{G}$ , the  $(u, v)$ -th entry of  $O_k$  counts the number of groundings of  $\phi$  in  $\mathcal{G}$  where  $U_1 = u$  and  $V_P = v$ . Therefore  $n_\phi(\mathcal{G}) = \sum(O_k(\phi))$ .
2. For an expected graph  $\tilde{\mathcal{G}}_z$ , the  $(u, v)$ -th entry of  $O_k$  counts the expected number of groundings of  $\phi$  where  $U_1 = u$  and  $V_P = v$ . Therefore  $n_\phi(\tilde{\mathcal{G}}_z) = \sum(O_k(\phi))$ .

*Proof.* We give the proof for clause 1, counting observed counts in an input graph  $\mathcal{G}$ . The argument for expected counts computed from an expected graph is exactly parallel.

Base case,  $k = 1$ . If  $\phi = \{\ell(U, V)\}$ , then the conjunction count is the number of pairs  $(u, v)$  such that (i) both groundings  $U = u$  and  $V = v$  satisfy all unary literals, and (ii)  $I_{\mathcal{G}}(\ell(U = u, V = v)) = 1$ . All and only such pairs have the entry  $\bar{\mathbf{A}}[u, v] = 1$  in the masked adjacency matrix associated with  $\ell(U, V)$ .

Case 1:  $(U = V) \notin EQ$ . Then the number of satisfying groundings is simply given by  $\sum(\bar{\mathbf{A}})$ .

Case 2:  $(U = V) \in EQ$ . Then the satisfying groundings are of the form  $U = u, V = u$ , so their count is given by the matrix trace of  $\bar{\mathbf{A}}$ , or equivalently  $\sum(\bar{\mathbf{A}} \circ I)$ . This establishes the base case.

Inductive Step: Assume the proposition holds for  $k$  and consider the matrix  $O_{k+1}$  computed by Equation (9). By the inductive hypothesis, the  $(u, v)$ -th entry of  $O_k$  counts the number of instantiations of length  $k$  between vertices  $u$  and  $v$  that satisfy  $\phi'$ . Now, the number of instantiations of length  $k + 1$  between  $u$  and  $w$  equals the number of instantiations of length  $k$  from vertex  $u$  to each vertex  $v$  that has  $\ell_{k+1}$  relation with  $w$ . The non-zero entries of column  $w$  of masked matrix  $\bar{\mathbf{A}}_{k+1}$  represent  $vs$  related by  $\ell_{k+1}$  to  $w$ . So,  $(u, w)$ -th entry of  $O_k \bar{\mathbf{A}}_{k+1}$  gives the number of instantiations between  $u$  and  $w$  satisfying the centered conjunction and all equality constraints except possibly  $U_1 = V_{k+1}$ . Therefore for the case where  $U_1 = V_{k+1} \notin EQ$ , the matrix  $O_{k+1}$  satisfies the inductive hypothesis. For the case where  $U_1 = V_{k+1} \in EQ$ , we observe that the number of instantiations of length  $k + 1$  from node  $u$  to  $u$  equals the  $(u, u)$  diagonal entry of  $O_k \bar{\mathbf{A}}_{k+1}$  or equivalently,  $(O_k \bar{\mathbf{A}}_{k+1}) \circ I$ . Thus the total number of satisfying groundings is given by  $\sum(O_{k+1}(\phi))$  in either case, which establishes the inductive hypothesis.  $\square$

### Extensions

- Our counting method is based on a **sorting algorithm**. A sorting algorithm is a procedure that takes a set of relationship literals

$$\ell_1(U_1, V_1), \dots, \ell_P(U_P, V_P)$$

and determines if there exists a permutation  $\pi$  such that the literals can be arranged to form a *chain conjunction*. Specifically, it seeks to satisfy the equality constraints given by

$$V_{\pi(i-1)} = U_{\pi(i)} \quad \text{for } i = 2, \dots, P.$$

### Example

Consider the following conjunction of relationship literals:

$$\begin{aligned} & \text{AdvisedBy}(\text{Student}, \text{Professor}), \\ & \text{TaughtBy}(\text{Course}, \text{Professor}), \\ & \text{Registered}(\text{Student}, \text{Course}) \end{aligned}$$

This set of relationships is not a chain conjunction because there is no permutation of the literals that satisfies the necessary chain equality constraints. However, we can transform this into a chain conjunction using the reverse relations:

$$\begin{aligned} & \text{AdvisedBy}(\text{Student}, \text{Professor}), \\ & \text{Teaches}(\text{Professor}, \text{Course}), \\ & \text{TakeCourse}(\text{Course}, \text{Student}) \end{aligned}$$

In this case, the literals can be rearranged to form a chain conjunction, satisfying the equality constraints. Here *Teaches* is the reverse of *TaughtBy* and *TakeCourse* is the reverse of *Registered*.

- Our algorithm and proof can be extended to the case of *nested conjunctions*, which have no crossing equalities. Say that two variable equalities  $U_{k_1} = V_{k_2}$  and  $U_{k_3} = V_{k_4}$  **cross** if  $k_1 < k_3 < k_2$  and  $k_2 < k_4$ . A nested chain is composed of centered chains, so our matrix multiplication algorithm can be used to recursively compute instance counts.

## A.8 Graph Realism Metric for evaluating Graph Generation

How to quantitatively assess generated graphs has been studied in recent papers O’Bray *et al.* [2022]; Thompson *et al.* [2022]; Shirzad *et al.* [2022]. The general approach proceeds in two stages:

1. For a graph  $\mathcal{G}$ , extract a real-valued **descriptor** vector  $\phi(\mathcal{G})$ .
2. Measure the similarity  $\mu(\mathcal{G}, \hat{\mathcal{G}})$  of an observed graph  $\mathcal{G}$  and a generated graph  $\hat{\mathcal{G}}$  by applying a distance/kernel on their real-valued descriptors vectors  $\phi(\mathcal{G})$  and  $\phi(\hat{\mathcal{G}})$ .

The similarity of a set of observed graphs and a set of generated graphs can be quantified as the similarity of their descriptor sets using Maximum Mean Distance (MMD). The SOTA descriptor function utilizes a *reference embedding network* GNN  $\mathcal{E}$ . The embedder  $\mathcal{E}$  is obtained from random weights and is therefore independent of any of the models under evaluation.

We adapt the SOTA GNN-based approach to the setting of learning from a single training graph  $D$  as follows. We compare the training graph to generated expected graphs  $\tilde{\mathcal{G}}_1, \dots, \tilde{\mathcal{G}}_m$ . An expected graph  $\tilde{\mathcal{G}}_i$  is generated by sampling a node embedding  $z_i$  from the prior distribution  $p(z)$ , then applying the decoder model eq. (5) to  $z_i$ . Following Thompson *et al.* [2022], we apply an embedder  $\mathcal{E}$  with random weights to the training graph resp. generated expected graphs to obtain embeddings  $e$  resp.  $\hat{e}_1, \dots, \hat{e}_m$ . The random GNN option does not require multiple training graphs. The message-passing mechanism of GNNs naturally extends to weighted

graphs, so we can apply the GNN embedder to expected graphs directly. To quantify the similarity of the generated embeddings  $\hat{e}_1, \dots, \hat{e}_m$  to the training graph embedding  $e$ , we utilize the MMD metric with a linear kernel, which is recommended by Thompson *et al.* [2022]. We used their code to compute the MMD metric results.

### A.9 Node Classification Baseline Methods

**Generalizable Graph Masked AutoEncoder (GiGa-MAE)** [Shi *et al.*, 2023] introduces a novel graph encoder based on aligning different node embeddings that respectively encode structural and feature information. **MVGRL** is an inductive self-supervised approach for learning representations of nodes and graphs by contrasting different structural views of graphs [Hassani and Khasahmadi, 2020].  **$\mathcal{G}^2\text{Pxy}$**  [Zhang *et al.*, 2023] generates proxy nodes to support classification. For all methods we used the authors’ code to train a node classifier and generate class labels.

### A.10 Impact of Rules on Training.

In this section we show that rule matching has a big impact on training, by in effect initializing the GGM model in a part of weight space that encodes the rule knowledge. After the VGAE+R model has encoded the background knowledge in its weights, it learns to optimize the other components, including the node label loss. This shows that rule matching is a strong regularizer that takes the network to a very different part of weight space compared to the baseline VGAE+ loss, which supports better generalization.

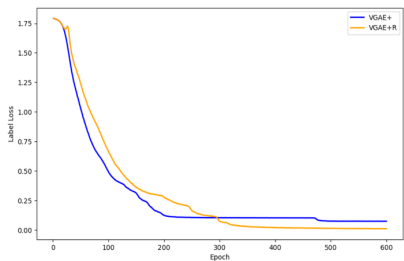


Figure 14: Label loss component for CiteSeer dataset during model training

Figure 14 shows the node label loss component of decoder training Equation (2) for the CiteSeer dataset. Rule matching adds a difficult new component to the VGAE+ objective, which initially causes a spike in the label loss component. The remainder of the supplement shows more loss curves with a similar pattern.

### Total Loss Curves

In this section, we present the total loss curves for both the VGAE+ and VGAE+R models across multiple datasets. While it is not possible to directly compare the loss values between the two models, the downward trend in loss during training for both models indicates successful convergence. Figures 15, 16, 17, 18, and 19 illustrate the total loss for each dataset, confirming the models’ progress throughout the training process.

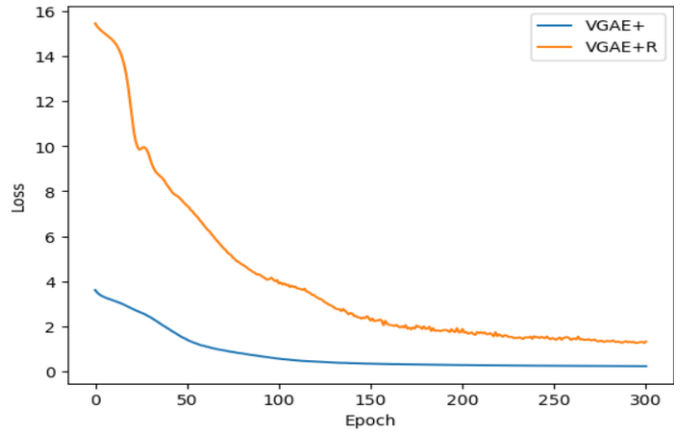


Figure 15: Total loss for Cora dataset during model training

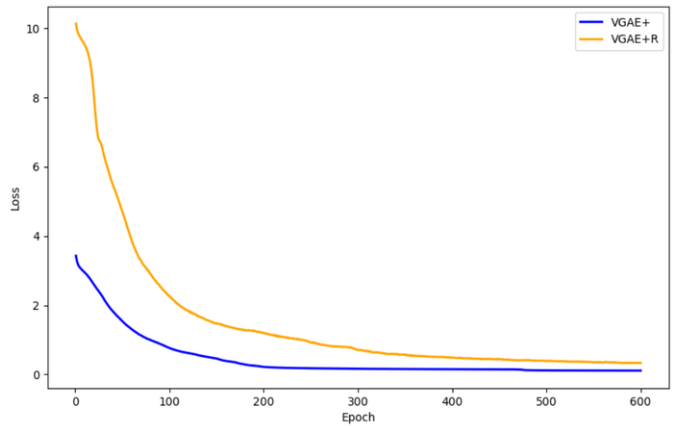


Figure 16: Total Loss for CiteSeer dataset during model training

### A.11 Label Loss Curves

In this section, we compare the label loss component for each model across different datasets. As shown in Figures 20, 21, 22, 23, and 24, the label loss for VGAE+R is generally lower than that for VGAE+ across most datasets, suggesting that VGAE+R may yield better accuracy in node classification.

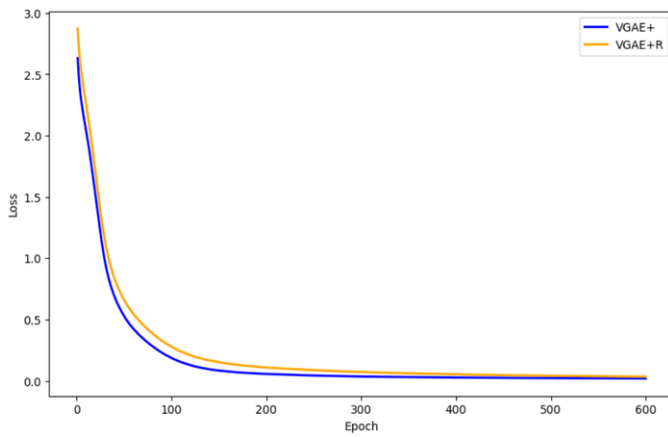


Figure 17: Total Loss for ACM dataset during model training

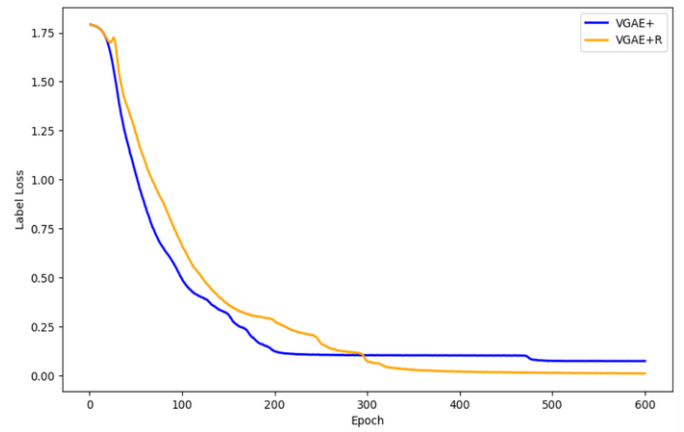


Figure 20: Label loss component for CiteSeer dataset during model training

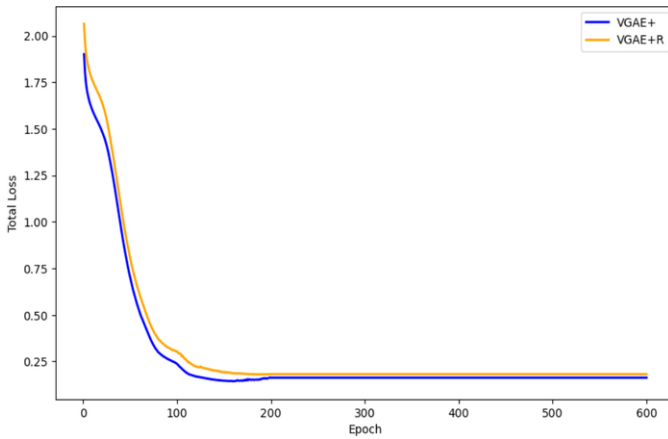


Figure 18: Total Loss for IMDb dataset during model training

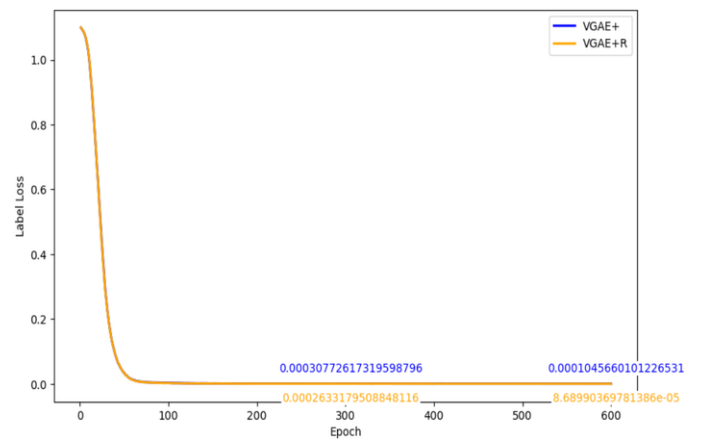


Figure 21: Label Loss component for ACM dataset during model training

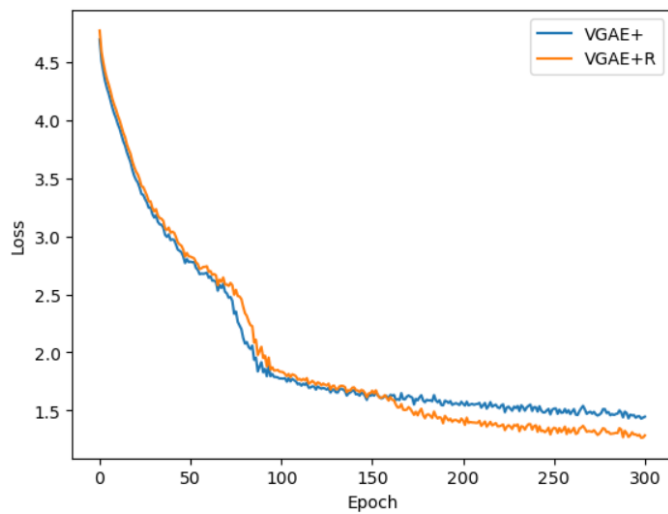


Figure 19: Total Loss for UW dataset during model training

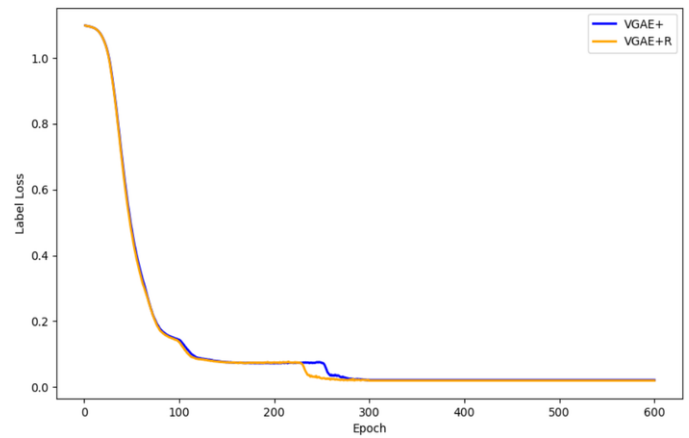


Figure 22: Label Loss component for IMDb dataset during model training

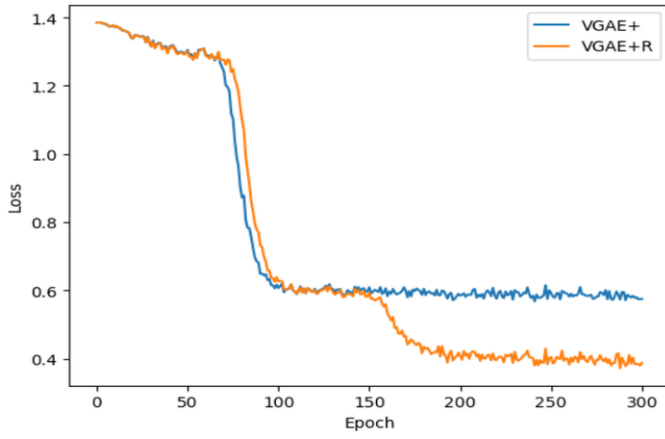


Figure 23: Label Loss component for UW dataset for person node type during model training

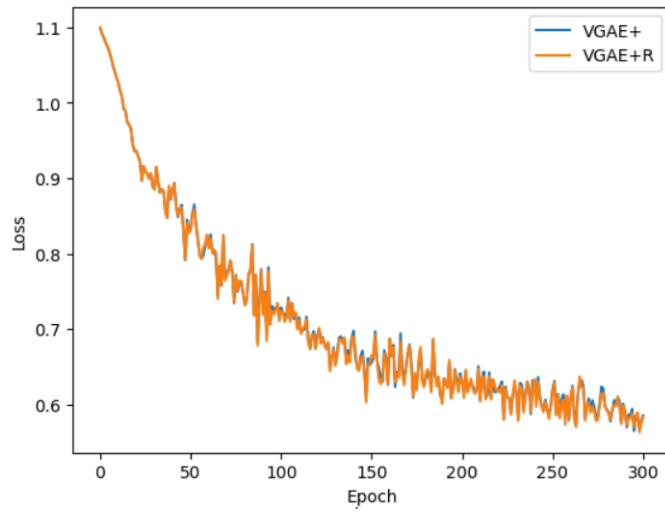


Figure 24: Label Loss component for UW dataset for course node type during model training