**Aalborg Universitet**

AALBORG UNIVERSITY

# Joint Link Prediction Via Inference from a Model

Naddaf, Parmis; Nejad, Erfaneh Mahmoudzaheh Ahmadi; Zahirnia, Kiarash; Jaeger, Manfred; Schulte, Oliver

*Published in:*
CIKM 2023 - Proceedings of the 32nd ACM International Conference on Information and Knowledge Management

*DOI (link to publication from Publisher):*
10.1145/3583780.3614941

*Publication date:*
2023

*Document Version*
Accepted author manuscript, peer reviewed version

Link to publication from Aalborg University

*Citation for published version (APA):*
Naddaf, P., Nejad, E. M. A., Zahirnia, K., Jaeger, M., & Schulte, O. (2023). Joint Link Prediction Via Inference from a Model. In *CIKM 2023 - Proceedings of the 32nd ACM International Conference on Information and Knowledge Management* (pp. 1877-1886). Association for Computing Machinery (ACM). https://doi.org/10.1145/3583780.3614941

**General rights**
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

**Take down policy**
If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from vbn.aau.dk on: March 21, 2026

# Joint Link Prediction Via Inference from a Model

Parmis Naddaf*
Erfaneh Mahmoudzadeh Ahmadi Nejad*
pnaddaf@sfu.ca
ema61@sfu.ca
Simon Fraser University
Burnaby, British Columbia, Canada

Kiarash Zahirnia
kzahirni@sfu.ca
Simon Fraser University
Burnaby, British Columbia, Canada

Manfred Jaeger
jaeger@cs.aau.dk
Aalborg University
Aalborg, Denmark

Oliver Schulte
oschulte@cs.sfu.ca
Simon Fraser University
Burnaby, British Columbia, Canada

## ABSTRACT

A Joint Link Prediction Query (JLPQ) specifies a *set* of links to be predicted, given another set of links as well as node attributes as evidence. While single link prediction has been well studied in literature on deep graph learning, predicting multiple links together has gained little attention. This paper presents a novel framework for computing JLPQs using a probabilistic deep Graph Generative Model. Specifically, we develop inference procedures for an inductively trained Variational Graph Auto-Encoder (VGAE) that estimates the joint link probability for any input JLPQ, without retraining. For evaluation, we apply inference to a range of joint link prediction queries on six benchmark datasets. We find that for most datasets and query types, joint link prediction via inference from a model achieves good predictive performance, better than the independent link prediction baselines (by 0.02-0.4 AUC points depending on the dataset).

## CCS CONCEPTS

• **Computing methodologies** → *Logical and relational learning*; **Neural networks**; *Learning latent representations*; • **Computing methodologies** → **Neural networks**; *Unsupervised learning*; Learning latent representations.

## KEYWORDS

Graph Representation Learning; Inference from a Model; Link Prediction; Graph Convolutional Networks

**ACM Reference Format:**
Parmis Naddaf, Erfaneh Mahmoudzadeh Ahmadi Nejad, Kiarash Zahirnia, Manfred Jaeger, and Oliver Schulte. 2023. Joint Link Prediction Via Inference from a Model. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, Birmingham, UK, 10 pages. https://doi.org/10.1145/3583780.3614941

*Both authors contributed equally to this research.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*CIKM '23, October 21-25, 2023, Birmingham, UK*
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3614941

## 1 INTRODUCTION: JOINT LINK PREDICTION

While link prediction is one of the main tasks in graph learning [10], to our knowledge work so far has almost exclusively focused on predicting a single link at a time. This paper addresses *joint link prediction*, where the task is to predict a set of target links from given evidence links. Since links in a graph are not independent of each other, optimal joint link prediction requires different techniques from predicting each target link independently. In the example of Figure 1, the probability that *both* Jennifer Aniston and Courteney Cox join a new show is higher than the product of the separate probabilities of them joining, because we know from other evidence links that they have collaborated in the past in the Friends series. This paper develops a query answering approach, where inference from a single trained model [19] is used to answer any user-specified JLPQ of the form: $\mathbb{P}(\textit{target links}|\textit{evidence links}, \textit{node attributes})$.

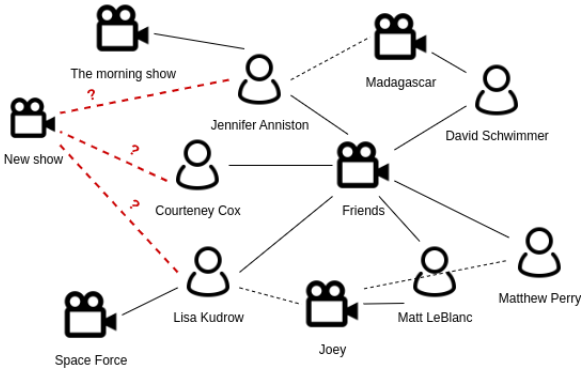



**Figure 1: Example of a joint link prediction. Given the main cast of the show Friends and some of other shows that they have played in as evidence, what is the probability of the female actors to play together in a new show. Target links are marked with "?". Solid links are specified as evidence in the JLP query, dashed links are not specified.**

**Motivation.** Joint link prediction queries are substantially more expressive than the previously studied link prediction tasks. First,

they allow users to choose the number of target links they select, rather than assuming a fixed prediction target size (i.e., one). Second, users can provide different kinds of evidence, depending on what is known, and use a single system for answering the query. In contrast, previous researchers have developed a separate customized method for different evidence types. Answering general relational queries, including joint link prediction, is a major use case for non-neural statistical-relational models, such as Markov Logic networks [4] and Probabilistic Soft Logic [13], which motivates our goal of expanding the prediction tasks to which neural models can be applied.

A single query answering system is important for deploying graph learning in a production environment where we do not know in advance which graph queries will be important, and users may not have the resources to build a customized machine learning solution for different query types. If the observed workload indicates that a particular query type is especially important for users, the system can be fine-tuned for it.

**Approach.** Figure 2 shows our system design. Our approach is a form of *domain-specific pre-training*: We learn a probabilistic Graph Generative Model (GGM) from data in a deployment domain. The model is used to answer queries with no further learning required. The deployment domain can be large (e.g., 1M nodes), whereas queries are typically small (e.g., a single link prediction query may involve two nodes and a $k$-hop neighborhood around them).

Specifically, we train a generative Variational Graph Auto-Encoder (VGAE) [10, 14] inductively, so that the VGAE can be applied to sets of target links that have different sizes and may involve unseen nodes. The generative probabilities over graphs (implicitly) define a conditional probability $P(target|evidence)$ for every JLPQ. The VGAE utilizes an encoder-decoder architecture with a Graph Neural Network (GNN) as an encoder. While VGAEs are not state-of-the-art for generating realistic graphs, they are useful for generating node embeddings for downstream tasks, such as link prediction [1]. Our main technical contribution are *approximate inference methods for estimating conditional joint link prediction probabilities from a trained VGAE.* In particular we introduce a novel importance sampling method that provides robust accurate predictions across multiple datasets.

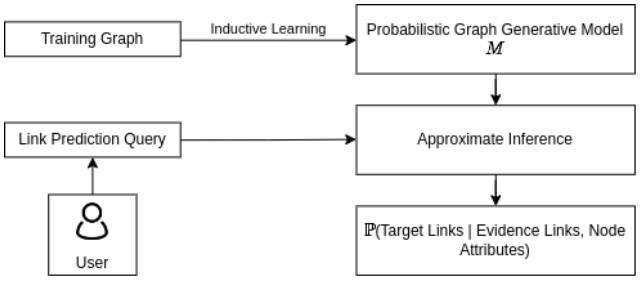



**Figure 2: After training a single GGM, a user query can be answered by applying the approximate inference methods described in this paper.**

**Evaluation.** We evaluate a variety of query types depending on whether 1) the target comprises multiple links or just a single link, and 2) the evidence comprises only links among test nodes (fully inductive), links among training and test nodes (semi inductive), or links among training nodes only (transductive). Our main baseline is to estimate link probabilities independently using SOTA single link methods. We find that the VGAE model achieves superior performance on many datasets and settings, especially inductive inference, and competitive on all, making it an effective generic methods for joint link prediction. We also benchmark our comparison methods on single link problems, and find that inference from the VGAE model is competitive with custom methods for single link prediction.

**Contributions.** Our main contributions are as follows.

- Introducing the task of joint link prediction.
- New inference methods for answering joint link prediction queries from a single trained GGM.
- An evaluation of our joint link prediction method on new prediction tasks.

*Paper Organization.* We review related work, then formally define joint link prediction queries. We describe inductive training of a VGAE based on the Evidence Lower BOund (ELBO) likelihood approximation. The main technical contribution is a description and evaluation of different inference methods for joint link prediction. We compare the empirical performance of joint link prediction with independent link prediction.

## 2 RELATED WORK

*Link Prediction Tasks.* For deep graph models, the most common setting is *single link completion*: predict a single target link given a set of known links, which typically include a large set of training links. Other recent variants include the following single link tasks. (1) Condition on the attributes of the two target nodes only [12]. (2) Condition on links among other test nodes only [22]. (3) Condition on links among training and test nodes [30]. Since our definition of joint link prediction allows arbitrary sets of evidence links, these are special cases. The only joint link prediction study we are aware is a note [9]. That work is restricted to temporal graphs and does not utilize inference from a probabilistic model.

*Inference from a Model.* Previous works on graph neural networks have focused on specific predictive tasks specified by the researcher. To our knowledge, inference from a generative model to estimate a large class of conditional probabilities is a novel application of neural graph learning.

*Graph Queries.* Powerful (non-probabilistic) graph query languages have been developed, such as Cypher and SPARQL [6, 8, 17]. Such graph queries typically return a set of nodes or links that satisfy a complex condition. In contrast, JLPQs probabilistic graph queries return a probability for the (non)existence of a set of links.

*Inductive Graph Training.* Inductive graph learning has been a major topic in recent research [11, 18, 28]. To support link prediction for nodes not seen during training, our query answering approach can be used with any inductive encoder-decoder graph neural network architecture, trained with the variational ELBO objective described below. To our knowledge, this is the *first use of the variational ELBO objective to support inductive link prediction.*

*CVAE for JLPGQs.* CVAEs are models for generating structured output [21], which makes them suitable for graph prediction. While they have been extensively applied to visual data, such as images

and videos, to our knowledge a CVAE has not previously been applied to graph data. Formally our JLPQ inference method can be viewed as using CVAE methods for inference. CVAEs are trained with the conditional ELBO training objective, for a fixed type of target and evidence. Since our goal is a general query answering system, we do not use the conditional ELBO.

*Graph Generative Models.* Our approach requires the following properties of a GGM. (1) Supports the computation/estimation of explicit graph probabilities. (2) Admits a conditional variant. (3) Applies to graphs of different sizes. As we show in this paper, VGAEs meet these requirements, so we base our experiments on them. Specifically, we employ the most recent VGAE designed for link prediction [1]. In section 7.2 we review other common GGMs with respect to these three criteria. We believe that developing GGMs so that they support answering joint link prediction queries is a fruitful new direction for GNNs.

*Structure-Conditioned Graph Generation* is a recently studied variant where the user specifies an induced subgraph (the evidence) and a conditional graph generation process completes it [5, 24, Sec.D.1]. Our definition of joint link prediction allows that neither evidence nor target links define a complete graph, which raises special challenges (see Section 5.2). In addition, our aim is to output an explicit probability over a joint link assignment, not output a graph. The fact that recent work has studied a related task beyond single link prediction supports our motivation for joint link prediction.

## 3 JOINT LINK PREDICTION QUERIES

JLP queries specify a set of links and attributes for a set of query nodes. The specification can involve a language as complex as first-order logic [4]. In this paper we introduce a relatively simple JLPGQ syntax—essentially, conjunctions of links and/or attributes—that suffices to express the prediction tasks that have been previously studied in graph learning.

*Data Format.* An attributed graph is a pair $G = (V, E)$ comprising a finite set of nodes $V$ and links $E$ where each node is assigned a $k$-dimensional attribute $\boldsymbol{x}_i$ with $k > 0$. An attributed graph with $N$ nodes can be represented by an $N \times N$ adjacency matrix $\boldsymbol{A}$ with $\{0, 1\}$ Boolean entries, together with an $N \times k$ node attribute matrix $\boldsymbol{X}$. Given a subset of nodes $V' \subseteq V$, the (induced) subgraph $G{\restriction}V'$ comprises the subset and all links among them: $G{\restriction}V' = (V', E')$ where $E' = E \cap (V' \times V')$.

### 3.1 Definition of JLP Queries

A relational random variable corresponds to either a node attribute or an adjacency. Let $\mathcal{A} = \{\boldsymbol{A}[u, v] : u \in V, v \in V\}$ be the set of **link variables**. A **joint link prediction query** is of the form

$$\mathbb{P}(\mathcal{A}^Y = \boldsymbol{y} | \mathcal{A}^E = \boldsymbol{e}, \boldsymbol{X}^E = \boldsymbol{x}) \tag{1}$$

where

- $\mathcal{A}^Y = \{\boldsymbol{A}[u_i, v_i] : i = 1, \ldots, m\}$ is the list of $m$ binary target links, each assigned a value $y_i \in \{0, 1\}$.
- $\mathcal{A}^E = \{\boldsymbol{A}[u_i, v_i] : i = 1, \ldots, e\}$ is the list of $e$ binary evidence links, each assigned a value $e_i \in \{0, 1\}$.
- $\boldsymbol{X}^E \in \mathbb{R}^{n \times k}$ denotes the node attributes of all the $n$ **query nodes** that appear in the query.

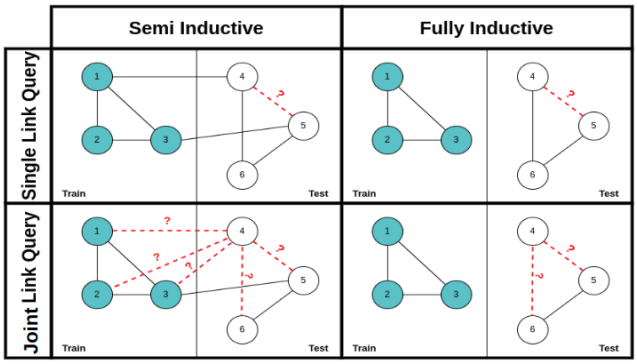



**Figure 3: Semi/fully inductive and single/joint link prediction queries.**

Given a partition of nodes into observed training nodes and unobserved test nodes, a query is **inductive** if a test node appears in the target.

### 3.2 Examples of Joint Link Prediction Queries

Figure 3 provides examples of important JLPQs. Single link prediction from the remaining graph is by far the most common link prediction setting, for instance in recommendation systems [30]. A **neighborhood query** targets the potential neighbors of a test node. In a recommendation setting, neighborhood prediction corresponds to predicting whether a user will select a set of items. In a social network setting, it corresponds to predicting a group of friends for a user.

We further distinguish query types according to how much they generalize from the training data. The **input graph** is one large graph $G_{in} = (V, E)$. Suppose the nodes in the graph are divided into training nodes $V_{tr}$ and inductive test nodes $V_{te}$. The **training graph** is the subgraph induced by the training nodes; that is, we remove the test nodes and their links. Similarly the **test graph** is the subgraph induced by the test nodes.

In a **fully inductive query**, target links are from the test graph. Evidence links comprise all test graph links that are not targets. In a **semi inductive query**, target links involve at least one test node. Evidence links comprise all links that are not targets (both training and test). Figure 3 illustrates fully and semi inductive queries.

## 4 VARIATIONAL GRAPH AUTO-ENCODER TRAINING

In this paper, we take the Variational Graph Auto-Encoder as our basic generative model for answering JLPQs. Let $\boldsymbol{Z}$ be an $N \times d$ matrix that represents latent node embeddings. In the VGAE model, links are generated independently given node embeddings:

$$p_\theta(\boldsymbol{A}|\boldsymbol{Z}) = \prod_{u,v} p_\theta(\boldsymbol{A}[u, v]|\boldsymbol{z}[u], \boldsymbol{z}[v]), \tag{2}$$

where $p_\theta : \mathbb{R}^d \times \mathbb{R}^d \to [0, 1]$ is a trainable **decoder**. The graph **encoder** $q_\phi(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{A})$ is implemented by a GNN that takes as input an attribute graph and returns latent node embeddings.

A VGAE is trained using the variational ELBO objective [10, 14]:

$$\mathcal{L}(\theta,\phi) = -E_{Z\sim q_\phi(Z|\boldsymbol{X},\boldsymbol{A})}\left[\ln p_\theta(\boldsymbol{A}|Z)\right] + KL\big(q_\phi(Z|\boldsymbol{X},\boldsymbol{A})||p(Z)\big). \tag{3}$$

In order to answer inductive queries involving unseen nodes, we train the VGAE without node IDs [10, Ch.5.1.1]. Equation (3) is the training objective used in our experiments.

# 5 MODEL INFERENCE FOR JOINT LINK PREDICTION

The *joint link prediction inference problem* is defined as follows.

- **Given** a single fixed graph generative model $M$, defining a distribution $\mathbb{P}_M$ over graphs,
- a joint link prediction query (Section 3.1) with target link set $\mathcal{A}^Y$ and evidence link set $\mathcal{A}^{\boldsymbol{E}}$, chosen by the user,
- **output** an estimate of

$$\mathbb{P}_M(\mathcal{A}^Y|\mathcal{A}^{\boldsymbol{E}},\boldsymbol{X}^{\boldsymbol{E}}) \equiv \mathbb{P}_M(\mathcal{A}^Y|\boldsymbol{E})$$

where we abbreviate the evidence links and attributes together as $\boldsymbol{E}$. According to the mixture model (2), the joint link prediction probability is given by

$$\mathbb{P}_M(\mathcal{A}^Y|\boldsymbol{E}) = \int_Z \mathbb{P}(\mathcal{A}^Y|Z)p(Z|\boldsymbol{E})dZ. \tag{4}$$

We describe three inference schemas for approximating the intractable integral (4). All three are evaluated in our experiments.

## 5.1 Inference Methods

In the following, let $\boldsymbol{Z} \in \mathbb{R}^{n\times d}$ be a set of node embeddings for the $n$ query nodes. The **evidence embeddings** are represented by a conditional Gaussian distribution, such that $p(\boldsymbol{z}[u]|\boldsymbol{E}) \sim N(\mu[u],\sigma[u])$ with mean and covariance $\mu[u],\sigma[u]$ for node $u$.

**Deterministic inference** computes the mean of the evidence embeddings, then applies the decoder to the mean to compute a target probability:

$$\mathbb{P}(\mathcal{A}^Y|\boldsymbol{E}) \approx \prod_{i=1}^{m} p(\boldsymbol{A}[u_i,v_i]|\mu[u_i],\mu[v_i]). \tag{5}$$

**Monte Carlo inference** samples evidence node embeddings, then applies the decoder to each sample. For every evidence link $[u,v]$ in $\boldsymbol{e}$, the evidence joint distribution over both node embeddings is given by:

$$\begin{aligned} &p(\boldsymbol{z}[u],\boldsymbol{z}[v]|\boldsymbol{E}) = p(\boldsymbol{z}[u]|\boldsymbol{E}) \times p(\boldsymbol{z}[v]|\boldsymbol{E}) \\ &\sim N(\mu[u],\sigma[u]) \times N(\mu[v],\sigma[v]). \end{aligned}$$

We now sample from the joint evidence distribution to estimate:

$$\begin{aligned} &\mathbb{P}(\mathcal{A}^Y|\boldsymbol{E}) \approx \frac{1}{S}\sum_{s=1}^{S}\prod_{i=1}^{m} p(\boldsymbol{A}[u_i,v_i]|\boldsymbol{z}^s[u_i],\boldsymbol{z}^s[v_i]) \\ &\boldsymbol{z}^s[u_i],\boldsymbol{z}^s[v_i] \sim p(\boldsymbol{z}[u_i],\boldsymbol{z}[v_i]|\boldsymbol{E}). \end{aligned} \tag{6}$$

We also examine **importance sampling** with respect to **proposal embeddings** $g(\boldsymbol{z}[u],\boldsymbol{z}[v]|\boldsymbol{E})$.

$$\begin{aligned} &\mathbb{P}(\mathcal{A}^Y|\boldsymbol{E}) \approx \frac{1}{S}\sum_{s=1}^{S}\prod_{i=1}^{m} p(\boldsymbol{A}[u_i,v_i]|\boldsymbol{z}^s[u_i],\boldsymbol{z}^s[v_i])\times \\ &\frac{p(\boldsymbol{z}^s[u_i],\boldsymbol{z}^s[v_i]|\boldsymbol{E})}{g(\boldsymbol{z}^s[u_i],\boldsymbol{z}^s[v_i]|\boldsymbol{E})} \\ &\boldsymbol{z}^s[u_i],\boldsymbol{z}^s[v_i] \sim g(\boldsymbol{z}[u_i],\boldsymbol{z}[v_i]|\boldsymbol{E}). \end{aligned} \tag{7}$$

Sohn et. al introduced the CVAE as the conditional version of a generative Variational Auto-Encoder (VAE) for estimating conditional probabilities for structured output [21]. They describe the three inference methods described above. In their terminology, the evidence embeddings $p(Z|\boldsymbol{E})$ are computed by a prior network. While their target application is image completion, CVAE inference is designed to generate structured output in general, hence is appropriate for graph prediction as well. In contrast to the CVAE, we use the inference methods with a single model trained independently of the query to be answered, whereas Sohn et al. train a model for a specific query type using the conditional ELBO. The next section shows how we can make the inference methods operational by computing evidence and proposal embeddings.

## 5.2 Computing evidence and proposal embeddings

Training a VGAE model $M$ provides a decoder $p_\theta$ and an encoder $q_\phi$ (Section 4). We employ the trained decoder for computing link probabilities given embeddings. To compute evidence embeddings, our approach is to apply the encoder to the evidence links/attributes. *The challenge is that the encoder assumes as input an adjacency matrix, or equivalently, a complete subgraph induced by the query nodes.* Inspired by previous work, we bridge the gap between a set of links and an (induced) subgraph by imputing link assignments missing from the evidence with a default value.

Figure 4 illustrates the problem and our solution. To define the issue formally, let $U$ be the set of $n$ query nodes, and let $\mathcal{A}_U$ be the set of $n^2$ link variables involving nodes from $U$. We write $\mathcal{A}^{\overline{E}} = \mathcal{A}_U - \mathcal{A}^{\boldsymbol{E}}$ for the set of unspecified links. The evidence links define an induced subgraph if and only if $\mathcal{A}^{\overline{E}} = \emptyset$. The evidence embeddings can in principle be computed as the marginalization over all subgraph completions of the evidence links:

$$p(Z|\mathcal{A}^{\boldsymbol{E}},\boldsymbol{X}^{\boldsymbol{E}}) = \sum_{\mathcal{A}^{\overline{E}}} p(Z|\mathcal{A}^{\boldsymbol{E}},\mathcal{A}^{\overline{E}},\boldsymbol{X}^{\boldsymbol{E}}) \times p(\mathcal{A}^{\overline{E}}|\mathcal{A}^{\boldsymbol{E}},\boldsymbol{X}^{\boldsymbol{E}}) \tag{8}$$

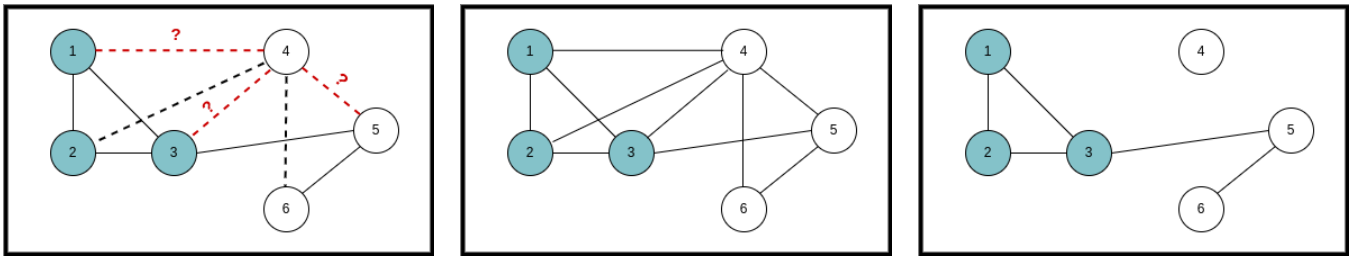
The difficulties with Equation (8) are that i) it involves an intractable exponential sum, and ii) it is circular in that the probability of subgraph completions given the evidence links is another joint link prediction problem. We propose to use 0 as a default value for **evidence embeddings**, that is, setting unspecified links to false:

$$p(Z|\mathcal{A}^{\boldsymbol{E}},\boldsymbol{X}^{\boldsymbol{E}}) \approx q_\phi(Z|\boldsymbol{A}^{\boldsymbol{E},\boldsymbol{0}},\boldsymbol{X}^{\boldsymbol{E}}) \tag{9}$$

where the encoder $q_\phi$ is applied to the adjacency matrix $\boldsymbol{A}^{\boldsymbol{E},\boldsymbol{0}}$ defined as follows: if $\mathcal{A}^{\boldsymbol{E}}$ specifies an assignment $\mathcal{A}[u,v] = e_i$, then $\boldsymbol{A}^{\boldsymbol{E},\boldsymbol{0}} := e_i$; otherwise $\boldsymbol{A}^{\boldsymbol{E},\boldsymbol{0}} := 0$. See Figure 4c.

Two motivations for setting unspecified links to 0 are the following. 1) A message-passing encoder typically propagates information only between neighbors. Using the 0 default value thus corresponds to treating unspecified links as uninformative. 2) Since real-world graphs are sparse, the 0 assignment is close to the mode of the evidence-conditional graph distribution.

In work on single link prediction, a related issue arises in a train-test split setting, where we want to predict a single test link given a set of training link values, but without specifying values for other test links. In this case the standard approach is to set the values of unspecified test links to 0 [14], which is a special case of our




(a) JLP Query. (b) Graph for proposal embeddings. (c) Graph for evidence embeddings.

**Figure 4: Evidence and proposal embeddings are computed from the evidence and proposal subgraphs respectively. Target links are marked with "?". Solid links are specified as evidence in the JLP query, dashed links are not specified.**

| Dataset | Nodes | Links |
|---|---|---|
| Cora [14] | 2,708 | 5,429 |
| ACM [27] | 8,993 | 18,929 |
| IMDb [27] | 12,772 | 19,120 |
| CiteSeer [14] | 3,327 | 4,732 |
| Photo [12] | 7,650 | 238,162 |
| Computers [12] | 13,752 | 491,722 |

**Table 1: Statistics of the datasets.**

proposal. Another example is the application of the conditional Variational Auto-encoder to image completion, where unspecified pixels are set to 0 [21]. Our experiments below show that using 0 as a default value achieves satisfactory results with the Monte Carlo and deterministic inference methods (Section 6.4).

Importance sampling requires specifying **proposal embeddings**. In our experiments, we set unspecified links to 1, which allows more information propagation through message passing:

$$g(Z|\mathcal{A}^{\boldsymbol{E}}, \boldsymbol{X}^{\boldsymbol{E}}) = q_{\phi}(Z|\boldsymbol{A}^{\boldsymbol{E},\mathbf{1}}, \boldsymbol{X}^{\boldsymbol{E}}) \tag{10}$$

where the adjacency matrix $\boldsymbol{A}^{\boldsymbol{E},\mathbf{1}}$ is defined as follows: if $\mathcal{A}^{\boldsymbol{E}}$ specifies an assignment $\mathcal{A}[u, v] = e_i$, then $\boldsymbol{A}^{\boldsymbol{E},\mathbf{1}} := e_i$; otherwise $\boldsymbol{A}^{\boldsymbol{E},\mathbf{1}} := 1$. The proposal distribution Equation (10) is combined with the approximate prior Equation (9) through Equation (7).

# 6 EMPIRICAL EVALUATION

We describe our experimental design and discuss our findings.

## 6.1 Experimental Design

We describe benchmark datasets used, different test queries, evaluation metrics and the query answering methods compared. Our GitHub repository provides a PyTorch implementation and datasets.[1]

*Datasets.* We utilize the datasets from previous studies of GGMs [12, 14, 27] ( Table 1). Cora, ACM, and CiteSeer are citation networks, IMDb is a movie dataset, and Photo and Computers are co-occurrence networks based on Amazon data.

[1]https://github.com/parmisnaddaf/Joint-Link-Prediction

*Data Preprocessing.* Following previous link prediction studies [14], we add self-loops and make all links undirected (i.e., if the training data contains an adjacency, $v \rightarrow u$, it also contains $u \rightarrow v$.)

## 6.2 Test Query Design

Since our input data are undirected graphs, we enforce the constraint that if a query target contains a link indicator $A[u, v]$, the evidence does not contain $A[v, u]$. We partition the nodes in the input graph into 1) 70% training nodes 2) 20% inductive test nodes and 3) 10% validation nodes. Our test queries are based on the query types described in Section 3.2, which differ according to whether they feature unseen nodes in the evidence/target.

**Transductive** All query nodes are training nodes. The evidence includes all links between training nodes that are not target links.

**Fully Inductive** All query nodes are test nodes. The evidence includes all links between test nodes that are not target links.

**Semi inductive** Each target link connects at least one test node. The evidence set comprises all links from the input graph that are not target links.

Note that in all transductive and inductive settings, the encoder sees only node attributes, not node IDs (Section 4), which is different from how some of the baseline methods are applied in the literature.

*Single Link Prediction Queries.* We randomly select 100 positive and 100 negative links that meet the criteria for the applicable setting. For example, in the semi inductive setting, we randomly select 100 positive and 100 negative links as target links for the query, such that at least one of the linked nodes is in the test set.

*Joint Link Prediction Queries.* We randomly select a set of 100 nodes as target nodes according to the applicable setting (i.e., target nodes are training nodes in transductive queries, and test nodes in semi/fully inductive queries). Then we ask the method to predict the neighbors of the target nodes as follows (cf. Figure 3). In the transductive and semi inductive settings, for each target test node with $l$ neighbors in the input graph, the target set comprises the $l$ links to its neighbors, and also $l$ links to randomly selected nodes that are not neighbors. In the fully inductive setting, for each target test node with $l$ test node neighbors, the target set comprises the $l$ links to its neighbors, and also $l$ links to randomly selected test nodes that are not neighbors.

*Metrics.* We discuss our evaluation metrics for single and joint link queries respectively.

**Single Link Metrics** We use AUC (Area Under Curve) and AP (Average Precision), standard metrics for class-imbalanced binary labels. AUC measures the area underneath the receiver operating characteristic (ROC) curve and AP calculates the average of the precisions over all possible classification thresholds [2]. We report the mean of each metric over all single link queries.

**Joint Link Metrics** For each joint link prediction query, we apply the same metrics, but on a per-query basis, that is, evaluated on the target links in each query. We report the mean of each metric over all joint link queries.

## 6.3 Query Answering Comparison Methods

**VGAE-Based Methods** We evaluate model inference based on three VGAE architectures trained with the ELBO (3). As a strong *decoder*, we utilize a non-linear node transformation function, Stochastic Block Model (SBM), [16] which is defined as:

$$p_\theta(\boldsymbol{z}[u], \boldsymbol{z}[v]) = \boldsymbol{z}[u]^\top \Lambda \boldsymbol{z}[v]$$

where $\Lambda \in R^{d \times d}$ is the block matrix for $d$ blocks in the Stochastic Block Model [10, Sec.8.3].

We compare three well-known *encoder* GNN architectures. All encoders have 2 layers and the output dimension is 128.

- **VGAE-GCN** Graph Convolutional Neural Network is a popular encoder [15].
- **VGAE-GAT** Graph Attention Networks adds link attention weights to graph convolutions [23].
- **VGAE-GIN** The Graph Isomorphism Network is a type of GNN that includes aggregation and combination [25].

We used the same VGAE implementation as [14], adding a recent normalization technique for link prediction [1]. We trained our models using 100 epochs, which was sufficient to ensure convergence of the training loss. The latent embedding dimension was set to 128 [14]. The GCN encoder is a two-layer GCN [64, 128]. The GAT encoder is also a two layer GAT with dimensions [32,128]. For each GAT layer, we used 4 heads as suggested by the original paper [23]. GAT layers concatenate embeddings for each head so we chose a smaller initial embedding dimension. For the GIN encoder we used maximum as the aggregator type.

*Inference methods.* We give results for deterministic prediction (Eq. (5)), Monte Carlo (Eq. (6)) with 30 samples, and Importance Sampling (Eq. (7)) with 30 samples.

**Baselines.** For comparing joint and independent link prediction, we select baselines that are well established and represent a variety of approaches to single link prediction. Specifically, we examine four baselines with three different frameworks:

1) **Classifier methods** use an inductive encoder, then train an SBM link prediction model on the node embeddings as input. Comparing classification methods with deterministic inference evaluates the importance of end-to-end training of the link predictor.

- **Classify(Sage)** Uses the well-known GraphSage embeddings, trained with the unsupervised objective [11].
- **Classify(GAT)** Uses the GAT embedding system, trained with the same micro-$F_1$ score as in the original paper [23].

Since these methods are designed to generalize to unseen test nodes, we do not apply them in the transductive setting.

2) **SEAL** is a well-known method that was introduced for transductive link prediction [29]. We train SEAL inductively by omitting node ids. This is similar to the approach of GraIL [22], here used for homogeneous graphs rather than knowledge graphs. The basic idea of SEAL is to embed the subgraph around a target link. It therefore benefits from denser input graphs.

3) **DEAL** was introduced for both transductive and inductive cold-start link prediction tasks [12]. The distinguishing feature of the DEAL approach is that at test time, it uses only the node attributes for link prediction. This makes it a strong baseline for independent prediction since it is not sensitive to the presence of evidence links. We apply DEAL only in the transductive and fully inductive settings mentioned in the original paper.

*Independent Link Prediction.* We extend the single link methods to predicting multiple links by computing independent link probabilities for each target link given the evidence (with unspecified links set to 0), then multiplying them to approximate the joint link probability. In symbols:

$$\mathbb{P}(\boldsymbol{A}[u, v_1], ..., \boldsymbol{A}[u, v_m] | \boldsymbol{E}) \approx \prod_{i=1}^{m} \mathbb{P}(\boldsymbol{A}[u, v_i] | \boldsymbol{E}). \tag{11}$$

where $u$ is the base node in our neighborhood queries. This comparison evaluates the difference between predicting the target links jointly vs. predicting each target link independently.

## 6.4 Experimental Results

We discuss our empirical findings.

*6.4.1 Main Comparison Methods.* Table 2 compares our recommended joint link prediction method VGAE-GIN with importance sampling (IS) to independent joint link prediction baselines. VGAE-based inference does very well on AUC, in all generalization settings. It generalizes well with strong AUC scores in the fully inductive setting ranging from 0.82 to 0.96.

*Inductive Joint Link Prediction Queries.* Inductive queries (both fully and semi inductive) are the most challenging link prediction setting so we discuss them in the most detail. On the AUC metric, the VGAE method outperforms all baselines on every dataset, except for SEAL on Cora and DEAL on ACM. The Embed+Classify methods do worse than other baselines, typically below 0.6. The AUC of SEAL is much lower on ACM, Photo and Computers, which shows that its ability to handle inductive joint link prediction is highly data-dependent. Similarly its generalization ability from semi to fully inductive link prediction depends strongly on the data: excellent on Cora and good on Photo and Computers, and very poor on ACM, IMDb, and CiteSeer. This spread likely arises because test graphs vary in their density.

DEAL was designed for inductive link prediction and is indeed superior to SEAL on all datasets except for Cora. It is especially accurate on the node-heterogeneous datasets ACM and IMDb, where an important component of link prediction are node types.

On the AP metric, a similar overall performance pattern holds. In sum, we conclude that *model-based inference provides accurate*

| | | AUC | | | | | | AP | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Cora | ACM | IMDb | CiteSeer | Photo | Computers | Cora | ACM | IMDb | CiteSeer | Photo | Computers |
| **Transductive** | VGAE-GIN (IS) | **0.742** | 0.952 | 0.843 | 0.876 | **0.867** | 0.800 | **0.775** | 0.963 | 0.858 | 0.887 | 0.849 | 0.780 |
| | SEAL | 0.741 | 0.500 | 0.620 | 0.752 | 0.684 | 0.701 | 0.655 | 0.322 | 0.641 | 0.738 | 0.790 | 0.720 |
| | DEAL | 0.676 | **0.979** | **0.962** | **0.910** | 0.827 | **0.818** | 0.647 | **0.985** | **0.969** | **0.927** | **0.894** | **0.862** |
| **Semi** | VGAE-GIN (IS) | 0.660 | **0.909** | **0.842** | **0.793** | **0.793** | **0.785** | 0.696 | **0.926** | **0.869** | **0.814** | **0.771** | **0.765** |
| | SEAL | **0.693** | 0.681 | 0.830 | 0.679 | 0.506 | 0.450 | **0.761** | 0.502 | 0.849 | 0.711 | 0.658 | 0.627 |
| | Classify (GAT) | 0.491 | 0.699 | 0.632 | 0.470 | 0.453 | 0.380 | 0.599 | 0.748 | 0.708 | 0.584 | 0.481 | 0.458 |
| | Classify (Sage) | 0.543 | 0.535 | 0.467 | 0.512 | 0.528 | 0.507 | 0.643 | 0.652 | 0.619 | 0.642 | 0.566 | 0.542 |
| **Fully** | VGAE-GIN (IS) | **0.826** | 0.932 | **0.960** | **0.891** | **0.903** | **0.820** | **0.849** | 0.951 | **0.970** | **0.918** | **0.884** | 0.825 |
| | SEAL | 0.821 | 0.260 | 0.336 | 0.375 | 0.657 | 0.687 | 0.748 | 0.545 | 0.577 | 0.544 | 0.741 | 0.775 |
| | Classify (GAT) | 0.394 | 0.621 | 0.587 | 0.450 | 0.537 | 0.468 | 0.628 | 0.737 | 0.743 | 0.655 | 0.583 | 0.534 |
| | Classify (Sage) | 0.557 | 0.513 | 0.459 | 0.532 | 0.554 | 0.541 | 0.723 | 0.678 | 0.659 | 0.707 | 0.622 | 0.616 |
| | DEAL | 0.678 | **0.953** | 0.935 | 0.837 | 0.759 | 0.757 | 0.651 | **0.968** | 0.952 | 0.864 | 0.836 | **0.844** |

**Table 2: AUC and AP results for joint link prediction. IS = Importance Sampling**

| | | AUC | | | | | | AP | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Cora | ACM | IMDb | CiteSeer | Photo | Computers | Cora | ACM | IMDb | CiteSeer | Photo | Computers |
| **Transductive** | VGAE-GIN (IS) | **0.920** | 0.957 | 0.971 | **0.975** | **0.961** | **0.910** | 0.737 | 0.810 | 0.849 | 0.874 | 0.813 | 0.830 |
| | SEAL | 0.910 | 0.904 | 0.906 | 0.948 | 0.935 | 0.908 | **0.925** | 0.907 | 0.934 | 0.926 | **0.914** | **0.908** |
| | DEAL | 0.800 | **0.986** | **0.981** | 0.914 | 0.832 | 0.823 | 0.824 | **0.987** | **0.984** | **0.928** | 0.823 | 0.818 |
| **Semi** | VGAE-GIN (IS) | **0.938** | **0.973** | **0.922** | **0.947** | **0.930** | 0.886 | 0.750 | **0.892** | 0.795 | **0.798** | 0.729 | 0.637 |
| | SEAL | 0.756 | 0.666 | 0.854 | 0.627 | 0.924 | **0.972** | **0.829** | 0.737 | **0.851** | 0.748 | **0.931** | **0.976** |
| | Classify (GAT) | 0.667 | 0.870 | 0.796 | 0.731 | 0.796 | 0.835 | 0.676 | 0.866 | 0.786 | 0.754 | 0.800 | 0.837 |
| | Classify (Sage) | 0.561 | 0.593 | 0.492 | 0.504 | 0.552 | 0.444 | 0.526 | 0.605 | 0.527 | 0.511 | 0.551 | 0.481 |
| **Fully** | VGAE-GIN (IS) | **0.903** | **0.988** | **0.967** | **0.974** | **0.960** | **0.976** | **0.849** | 0.926 | 0.869 | **0.927** | 0.771 | 0.756 |
| | SEAL | 0.695 | 0.645 | 0.636 | 0.579 | 0.840 | 0.939 | 0.787 | 0.701 | 0.647 | 0.697 | **0.897** | **0.959** |
| | Classify (GAT) | 0.525 | 0.764 | 0.671 | 0.700 | 0.802 | 0.733 | 0.534 | 0.781 | 0.667 | 0.695 | 0.799 | 0.757 |
| | Classify (Sage) | 0.695 | 0.564 | 0.440 | 0.521 | 0.572 | 0.515 | 0.486 | 0.554 | 0.471 | 0.532 | 0.627 | 0.549 |
| | DEAL | 0.733 | 0.957 | 0.948 | 0.828 | 0.861 | 0.829 | 0.780 | **0.967** | **0.957** | 0.861 | 0.852 | 0.844 |

**Table 3: AUC and AP results for single link prediction. IS = Importance Sampling**

*joint link predictions on inductive queries, superior to independent link prediction, and generalizes well to unseen test nodes.*

*Transductive Joint Link Prediction Queries.* We discuss how VGAE-GIN (IS) compares to the best baselines from the inductive setting. SEAL's performance on transductive queries is worse than VGAE inference, as with the inductive queries. However, DEAL achieves competitive scores, for AUC better than VGAE-GIN (IS) on 4 out of 6 datasets. This confirms our choice of DEAL as a strong baseline for independent link prediction. It appears that transductive link prediction is relatively easier for DEAL because the node attribute distribution does not change between training and test sets.

*Inductive Single Link Prediction.* Table 3 shows the results for single link prediction. Consistent with the findings of [1], the VGAE-GIN (IS) method performs well for single link prediction, better than SEAL on 5 out of 6 datasets on AUC. The accuracy of SEAL improves with bigger graphs. As in joint link prediction, VGAE-GIN (IS) generalizes well, and DEAL performs well on fully inductive queries. The Embed+Classify methods do worse than VGAE on all datasets on the AUC metric. Our first conclusion is that while VGAE inference is good for single link prediction, the strong performance for joint link prediction is due to the ability of the generative model to capture dependencies between links, not because of its single link prediction performance. Our second conclusion is that the joint link prediction scores Table 2 of VGAE-GIN (IS) are good not only relative to the baselines, but also absolutely, as they are close to the single link prediction scores of Table 3.

*Transductive Single Link Prediction Queries.* As with joint link prediction queries, DEAL is competitive with VGAE-GIN (IS), although the ranking reverses on a number of datasets, such that VGAE-GIN (IS) achieves a higher AUC score on 4 out of 6. The AUC scores of SEAL improve markedly, which illustrates that SEAL does not generalize well, especially compared to the VGAE and DEAL.

In most settings, DEAL is the strongest competitor to VGAE joint link prediction. Because at test time, DEAL uses no links but only node attributes, its performance depends strongly on the dataset and the informativeness of the available node attributes for link prediction (see also the discussion in [12]). Overall, our findings show that VGAE joint link prediction is a good generic methods that works well across a range of datasets and evidence settings.

| | | AUC | | | | | | AP | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Cora | ACM | IMDb | CiteSeer | Photo | Computers | Cora | ACM | IMDb | CiteSeer | Photo | Computers |
| **Semi** | VGAE-GCN (DET) | 0.721 | 0.958 | 0.827 | **0.951** | 0.757 | 0.761 | 0.759 | 0.963 | 0.849 | **0.954** | 0.735 | 0.743 |
| | VGAE-GCN (MC) | 0.711 | 0.959 | 0.836 | 0.930 | 0.802 | 0.841 | 0.755 | 0.964 | 0.861 | 0.941 | 0.788 | 0.824 |
| | VGAE-GCN (IS) | 0.733 | 0.962 | 0.793 | 0.928 | **0.914** | **0.861** | **0.766** | 0.964 | 0.811 | 0.942 | **0.895** | **0.834** |
| | VGAE-GAT (DET) | 0.737 | 0.925 | 0.814 | 0.932 | 0.759 | 0.615 | 0.757 | 0.938 | 0.837 | 0.943 | 0.705 | 0.590 |
| | VGAE-GAT (MC) | **0.749** | 0.937 | 0.815 | 0.925 | 0.812 | 0.775 | 0.764 | 0.947 | 0.841 | 0.945 | 0.778 | 0.721 |
| | VGAE-GAT (IS) | 0.729 | **0.963** | **0.854** | 0.909 | 0.845 | 0.806 | 0.746 | **0.968** | **0.886** | 0.893 | 0.801 | 0.804 |
| | VGAE-GIN (DET) | 0.684 | 0.935 | 0.835 | 0.894 | 0.809 | 0.733 | 0.724 | 0.944 | 0.869 | 0.893 | 0.753 | 0.686 |
| | VGAE-GIN (MC) | 0.683 | 0.939 | 0.842 | 0.891 | 0.810 | 0.746 | 0.721 | 0.949 | 0.873 | 0.893 | 0.756 | 0.704 |
| | VGAE-GIN (IS) | 0.660 | 0.909 | 0.843 | 0.793 | 0.793 | 0.785 | 0.696 | 0.926 | 0.869 | 0.814 | 0.771 | 0.765 |
| **Fully** | VGAE-GCN (DET) | 0.639 | 0.959 | 0.860 | 0.923 | 0.783 | 0.786 | 0.718 | 0.969 | 0.893 | 0.949 | 0.781 | 0.787 |
| | VGAE-GCN (MC) | 0.631 | 0.960 | 0.835 | 0.929 | 0.817 | 0.858 | 0.711 | 0.971 | 0.877 | **0.952** | 0.848 | **0.851** |
| | VGAE-GCN (IS) | 0.621 | 0.963 | 0.700 | 0.897 | 0.885 | 0.842 | 0.714 | 0.964 | 0.778 | 0.925 | 0.867 | 0.825 |
| | VGAE-GAT (DET) | 0.670 | 0.946 | 0.830 | 0.898 | 0.749 | 0.683 | 0.747 | 0.962 | 0.875 | 0.930 | 0.732 | 0.687 |
| | VGAE-GAT (MC) | 0.691 | **0.967** | 0.830 | 0.922 | 0.799 | 0.822 | 0.756 | **0.974** | 0.875 | 0.946 | 0.790 | 0.802 |
| | VGAE-GAT (IS) | 0.630 | 0.966 | 0.872 | **0.953** | 0.823 | 0.816 | 0.724 | 0.973 | 0.910 | 0.926 | 0.804 | 0.799 |
| | VGAE-GIN (DET) | 0.612 | 0.965 | 0.901 | 0.923 | 0.791 | 0.781 | **0.944** | 0.966 | 0.922 | 0.842 | 0.764 | 0.767 |
| | VGAE-GIN (MC) | 0.683 | 0.964 | 0.897 | 0.729 | 0.793 | **0.873** | 0.707 | 0.966 | 0.920 | 0.861 | 0.767 | 0.782 |
| | VGAE-GIN (IS) | **0.826** | 0.932 | **0.960** | 0.891 | **0.903** | 0.820 | 0.696 | 0.849 | **0.969** | 0.901 | **0.883** | 0.825 |

**Table 4: Ablation study for joint link prediction: different encoders/inference methods. DET= deterministic, MC = Monte Carlo.**

| | | AUC | | | | | | AP | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Cora | ACM | IMDb | CiteSeer | Photo | Computers | Cora | ACM | IMDb | CiteSeer | Photo | Computers |
| **Semi** | All 1 + MC | **0.832** | 0.762 | 0.729 | 0.886 | 0.699 | **0.892** | **0.837** | 0.800 | 0.778 | 0.903 | 0.692 | **0.864** |
| | Softmax of (all 1 + MC, all 0 + MC) | 0.519 | 0.102 | 0.231 | 0.261 | 0.510 | 0.527 | 0.604 | 0.397 | 0.473 | 0.460 | 0.494 | 0.500 |
| | Softmax of (IS, all 0 + MC) | 0.825 | 0.766 | 0.724 | **0.904** | **0.898** | 0.883 | 0.835 | 0.803 | 0.775 | **0.914** | **0.876** | 0.856 |
| | VGAE-GIN (IS) | 0.660 | **0.909** | **0.843** | 0.793 | 0.793 | 0.785 | 0.696 | **0.926** | **0.869** | 0.814 | 0.771 | 0.765 |
| **Fully** | All 1 + MC | 0.692 | 0.723 | 0.757 | 0.822 | 0.872 | **0.861** | **0.756** | 0.804 | 0.820 | 0.875 | 0.860 | **0.861** |
| | Softmax of (all 1 + MC, all 0 + MC) | 0.477 | 0.053 | 0.114 | 0.155 | 0.527 | 0.463 | 0.615 | 0.403 | 0.439 | 0.464 | 0.542 | 0.487 |
| | Softmax of (IS, all 0 + MC) | 0.679 | 0.705 | 0.755 | 0.836 | 0.861 | 0.841 | 0.741 | 0.795 | 0.821 | 0.885 | 0.849 | 0.843 |
| | VGAE-GIN (IS) | **0.826** | **0.932** | **0.960** | **0.891** | **0.903** | 0.820 | 0.696 | **0.849** | **0.969** | **0.901** | **0.883** | 0.825 |

**Table 5: Ablation study for joint link prediction: combining evidence and proposal embeddings.**

*6.4.2 Ablation Studies.* Table 4 examines the performance of VGAE methods with the different graph encoders and inference methods discussed so far. All VGAE-based methods do well compared to independent link prediction. This is evidence that inference from a GGM is a fundamentally sound approach. Within the VGAE-methods, there is no method with universally best empirical performance. Generally both Monte Carlo (MC) and IS score higher than the deterministic (DET) versions, especially on the Photo and Computers datasets, which shows the value of a probabilistic model. Importance sampling is reliable, and on some datasets achieves big improvements, especially on the bigger and denser Photo and Computers datasets. Both the GAT and the GIN encoders give good results, but GIN tends to score even better in the challenging fully inductive setting. *We therefore recommend IS inference with the GIN architecture as the default model for VGAE-based joint link prediction.*

*Combining Evidence and Proposal Embeddings.* Importance sampling can be viewed as combining the all 0 evidence and all 1 proposal embeddings. We evaluated also the following different approaches to combining them:

**All 1 + MC** : Set all unspecified links to 1 and apply Monte Carlo inference (6).

**Softmax of (all 1 + MC, all 0 + MC)** : Apply Monte Carlo to all 0 and all 1 default embeddings and combine then with softmax.

**Softmax of (IS, all 0 + MC)** : This is equivalent to doing importance sampling (7) with all 1/all 0 proposal embedings, and combining the results with softmax.

Table 5 shows the result for these different approaches. Overall, our recommended VGAE-GIN (IS) method outperforms the ablations. In the semi inductive setting, setting all unspecified edges to 1 is a simple method that scores higher than our recommend method on 3 out of 6 datasets. The differences are quite high. For example, the Cora difference is 0.172, but on ACM it is -0.147. This is further evidence that adding links helps improve information flow between links.

In the fully inductive setting VGAE-GIN (IS) outperforms the all 1 + MC ablation on all datasets, by as much as 0.227 AUC points on ACM. The pattern for the softmax of VGAE-GIN (IS) with all 0 + MC is similar. The softmax combination of the decoder outputs from all 1 and all 0 embeddings is very poor.

Overall, our conclusion is that there are several viable options for inference from a VGAE model, but our importance sampling method is the most robust across datasets and generalization settings.

*Computational Cost.* The models considered in this paper are fast to train, please see the analysis in the original papers. Sampling incurs computational overhead at test time. For example for joint link prediction on the most demanding dataset Computers, Importance Sampling takes 22.67 seconds per query on average, Monte Carlo takes 13.19 seconds, and Deterministic 2.78 seconds. Inference will be even faster assuming the evidence is restricted to a relevant small subset of the input graph (e.g. GNNs are typically applied to a $k$-hop neighborhood of the target links).

# 7 DISCUSSION: ALTERNATIVES AND LIMITATIONS

Joint link prediction is a challenging new problem; we believe that exploring different approaches is a fruitful direction for future research. In this section we discuss the pros and cons of some different approaches in addition to those evaluated in our experiments already. We first describe alternatives within the VGAE-based system, second the potential of other GGMs.

## 7.1 Other VGAE-based approaches

*Graph Encoders.* New developments in Graph Neural Networks, such as graph transformers [20], can be leveraged for JLP queries by replacing the encoder module.

*Inference Methods.* Instead of imputing unspecified links with 0/1 values to compute embeddings, a principled alternative is to view unspecified links as discrete link type: use a discrete link attribute with three possible values $0, 1, *$ where * stands for "unspecified". Then train a multi-relational encoder to compute node embeddings with three-valued links. The challenge is how to train the three-valued graph encoder, since the original training data comprises 0/1 links only. We investigated a BERT-style approach [3], where we randomly masked links with a special * token and trained an encoder to reconstruct the masked links. We found that the test query performance on the three-value approach was poor. For example, the average AUC score on Cora for single link prediction with masking, taken over the different VGAE-methods, was 0.750, compared to 0.938 for VGAE-GIN (IS) without masking (Table 3).

For extending single link methods for joint link prediction, an alternative to predicting links independently as in Equation (11) is to generate the target links in order using single link probabilities at each step. This is similar to using an auto-regressive conditional generation method (see next section). The difficulty is that the resulting joint probability will be highly sensitive to the chosen node ordering. For example, if we start with an incorrect link assignment $\boldsymbol{A}[u, v_1] := 1$ based on the single link method, the incorrect link may lead to another wrong decision for the next target link.

A limitation of our evaluation is that we considered only homogeneous graphs with a single link type. The deterministic and Monte Carlo can be extended straightforwardly to knowledge graphs using a relational VGAE model. For importance sampling, further research is required to extend the idea of an all 1 proposal embedding to multiple link types. Multi-relational joint link prediction can be compared to recent methods for inductive link prediction in knowledge graphs [7].

## 7.2 Other Graph Generative Models

We discuss whether several well-known GGMs support answering JLPQs, with reference to the criteria of section 2. The models are succinctly reviewed by Hamilton, except for graph diffusion [24]. GraphGANs do not provide explicit graph probabilities. GraphVAEs require graph matching to estimate probabilities for subgraphs, and are not designed for query target graphs much smaller than the training graph. Auto-regressive models provide explicit graph probabilities for graphs of different sizes. As the work on structure-conditioned graph generation has shown, they can be applied to evidence subgraphs [5]. A major issue is that auto-regressive methods require a node ordering and JLPQ inference is likely to be sensitive to the chosen ordering (see previous section). Faez et al. utilize random node orderings to generate graph completions. For graph diffusion [5], Vignac et al. describe a variant that conditions the diffusion process on a subgraph [24, Sec.D.1]. In their method, the subgraph is constrained to only known nodes. The computation of exact data likelihoods in diffusion models, in general, can be a challenging tasks [24, Sec.C], [26, Sec.4.3].

# 8 CONCLUSION AND FUTURE WORK

A Joint Link Prediction Query (JLPQ) asks for the probability of a target set of links, given an evidence set of links and node attributes. Supporting inference to answer JLPQs is a new use case for a Graph Generative Model (GGM). Such a query answering system facilitates applying graph prediction in a production environment where multiple users pose a range of queries to be answered. In this paper we showed how inference from a trained Variational Graph Auto-Encoder (VGAE) model can be used to answer JLPQs, in zero-shot manner without retraining the model. The key issue is how to apply a graph encoder when some links are unspecified by the query. The most reliable inference methods was based on importance sampling, where the base distribution sets unspecified links to 0 (absent) and the proposal distribution sets unspecified links to 1 (present). We carried out an empirical evaluation on six benchmark datasets and a range of queries and observed that the application of joint link prediction utilizing a single VGAE yielded higher accuracy compared to independent link prediction methods that are based on single link approaches.

*Future Work.* Combining the ability of DEAL to leverage node attributes with the ability of our VGAE model to leverage evidence links is a promising extension. Other GGMs, especially auto-regressive and diffusion models, have the potential to support powerful joint link prediction via inference, but require research challenges to be addressed. Showing that the greater expressive power of these GGMs can lead to improved joint link prediction over our strong VGAE baseline is a fruitful direction for future research. Another valuable direction is to apply to inference from a model to finding the most likely joint link assignment for a set of target links.

# REFERENCES

[1] Seong Jin Ahn and MyoungHo Kim. 2021. Variational Graph Normalized AutoEncoders. *In Proceedings of the ACM International Conference on Information & Knowledge Management*, 2827–2831.

[2] Jesse Davis and Mark Goadrich. 2006. The Relationship between Precision-Recall and ROC Curves. *In Proceedings of the 23rd International Conference on Machine Learning*, 233–240.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, MN, USA, 4171–4186.

[4] Pedro Domingos and Daniel Lowd. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence.* Morgan and Claypool Publishers.

[5] Faezeh Faez, Negin Hashemi Dijujin, Mahdieh Soleymani Baghshah, and Hamid R Rabiee. 2022. SCGG: A deep structure-conditioned graph generative model. *Plos one* 17, 11 (2022), e0277887.

[6] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. 2018. Cypher: An Evolving Query Language for Property Graphs. In *Proceedings of the 2018 International Conference on Management of Data* (Houston, TX, USA) *(SIGMOD '18)*. Association for Computing Machinery, New York, NY, USA, 1433–1445.

[7] Mikhail Galkin, Etienne Denis, Jiapeng Wu, and William L. Hamilton. 2022. NodePiece: Compositional and Parameter-Efficient Representations of Large Knowledge Graphs. *In Proceedings of the International Conference on Learning Representations*.

[8] Michael Galkin, Zhaocheng Zhu, Hongyu Ren, and Jian Tang. 2022. Inductive Logical Query Answering in Knowledge Graphs, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.). *In Proceedings of the Neural Information Processing Systems* 35, 15230–15243.

[9] Kyle Griswold, Seokho Hong, and N. Arivazhagan. 2015. Joint Link Prediction in Temporal Networks.

[10] William L Hamilton. 2020. *Graph representation learning.* Morgan & Claypool Publishers. 1–159 pages.

[11] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *In Proceedings of the 31st International Conference on Neural Information Processing Systems*, 1025–1035.

[12] Yu Hao, Xin Cao, Yixiang Fang, Xike Xie, and Sibo Wang. 2020. Inductive Link Prediction for Nodes Having Only Attribute Information. *Structure* 3, 4 (2020), 5.

[13] Bert Huang, Angelika Kimmig, Lise Getoor, and Jennifer Golbeck. 2012. Probabilistic soft logic for trust analysis in social networks. *In Proceedings of the International Workshop on Statistical Relational Artificial Intelligence (StaRAI 2012)*.

[14] Thomas N Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. *In Proceedings of the Neural Information Processing Systems Workshop on Bayesian Deep Learning* (2016).

[15] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. *In Proceedings of the International Conference on Learning Representations*.

[16] Nikhil Mehta, Lawrence Carin Duke, and Piyush Rai. 2019. Stochastic blockmodels meet graph neural networks. *In Proceedings of the International Conference on Machine Learning*, 4466–4474.

[17] Eric Prud'hommeaux and Andy Seaborne. 2007. SPARQL query language for RDF. (01 2007).

[18] Ryan A Rossi, Rong Zhou, and Nesreen K Ahmed. 2018. Deep inductive graph representation learning. *In Proceedings of the IEEE Transactions on Knowledge and Data Engineering* 32, 3 (2018), 438–452.

[19] Stuart Russell. 2015. Unifying logic and probability. *Commun. ACM* 58, 7 (2015), 88–97.

[20] Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J. Sutherland, and Ali Kemal Sinop. 2023. Exphormer: Scaling Graph Transformers with Expander Graphs. https://openreview.net/forum?id=8Tr3v4ueNd7

[21] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. *In Proceedings of the Neural Information Processing Systems* 28 (2015), 3483–3491.

[22] Komal K. Teru, Etienne G. Denis, and William L. Hamilton. 2020. Inductive Relation Prediction by Subgraph Reasoning. *In Proceedings of the 37th International Conference on Machine Learning*, Article 876, 10 pages.

[23] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *In Proceedings of the International Conference on Learning Representations* (2018).

[24] Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. 2023. DiGress: Discrete Denoising diffusion for graph generation. *In Proceedings of the Eleventh International Conference on Learning Representations*.

[25] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks? *In Proceedings of the International Conference on Learning Representations*.

[26] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2022. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796* (2022).

[27] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. *In Proceedings of the Neural Information Processing Systems* 32 (2019).

[28] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. 2020. GraphSAINT: Graph Sampling Based Inductive Learning Method. *In Proceedings of the International Conference on Learning Representations*.

[29] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. *In Proceedings of the Neural Information Processing Systems* 31 (2018).

[30] Muhan Zhang and Yixin Chen. 2020. Inductive Matrix Completion Based on Graph Neural Networks. *In Proceedings of the International Conference on Learning Representations*.