Deep Variational Inference for Inductive Link Prediction

Oliver Schulte [1], Parmis Naddaf [1], Xia Hu [1], Manfred Jaeger [2]

[1]School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
[2]Computer Science Department, Aalborg University, Aalborg Øst, Denmark
[1]{oschulte,pnaddaf,xia_hu}@sfu.ca, [2]jaeger@cs.aau.dk

Abstract

This paper describes a work in progress on a new theoretical framework that leverages variational inference for inductive link prediction. Inductive link prediction (ILP) estimates the probability that one or more test nodes not observed during training are linked to other nodes. Different ILP tasks arise from conditioning on different available evidence, such as node attributes and/or the existence of other links. Recent research has developed customized models for different ILP tasks; this paper describes a unified framework for solving ILP tasks based on an inductively trained probabilistic generative graph model, specifically a Variational Graph Auto-Encoder (VGAE). An inductive VGAE defines a distribution over adjacencies containing both training and test nodes, and therefore—implicitly—a conditional link probability, which can be approximated by variational inference with a conditional ELBO. We show how, given a trained VGAE and an ILP query, a conditional variational auto-encoder can be constructed dynamically that approximates the conditional ELBO *without* retraining on data. The variational framework allows us to address a wide class of ILP tasks with a single VGAE, including the standard single-link probabilities, as well as useful new joint link probabilities (e.g. completing the neighbourhood of a target test node).

Introduction: Inductive Link Prediction

Inductive link prediction is the task of predicting the existence of one or more links involving test nodes that are not observed during training. This paper describes a new unifying framework for ILP problems. It describes a complete approach but is a work in progress in that we have not yet completed an empirical evaluation.

ILP methods estimate conditional *query probabilities* that specify the probability of one or more *target links* that involve at least one test node, given conditioning *evidence* about node attributes/features and/or the existence of other links. Different ILP variants have been studied in recent works, including the following evidence settings for predicting the existence of a single target link. (1) Condition on the attributes of the connected test nodes (Hao et al. 2020). (2) Condition on links among other test nodes only (Teru, Denis, and Hamilton 2020). (3) Condition on links between both training and test nodes (Zhang and Chen 2019).

Previous research has developed a customized method for each ILP query. This paper introduces a new method that addresses a wide class of ILP problems in a unifying framework based on a single probabilistic generative graph model. Our query answering approach provides a form of classic *inference from a model* (Russell 2015): After learning a domain model from data, the model is used to answer queries with no further data access required. Figure 1 summarizes our design.

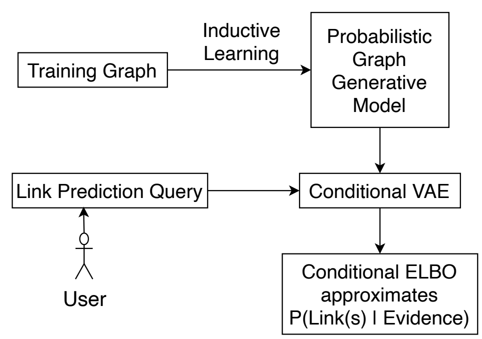


Figure 1: After training a single model, user queries can be answered by dynamically transforming a Conditional Variational Auto-Encoder from a probabilistic generative model.

A query answering approach has two motivations.

Use Cases It is difficult to determine in advance which ILP queries will be important for a user, so a single system that supports a wide variety is helpful. Some applications may require targets more complex than the traditional single link, such as the complete neighborhood of a test node. Likewise, the available evidence may vary widely, even on the same data set. Ease of use and wide applicability make inference from a single model useful, even if on some queries inference from a model is less accurate than a system custom-built for that query.

Evaluating Generative Models Evaluating the quality of a graph generative model is a difficult problem (O'Bray et al. 2021). Our framework allows the strengths and weaknesses of an inductively trained generative graph model to be assessed by its predictive performance for a variety of queries.

Approach We train a generative VGAE (Kipf and Welling 2016; Hamilton 2020) inductively, meaning that the VGAE defines probabilities over graphs of different sizes (Hamilton, Ying, and Leskovec 2017). Our main technical contribution is to implement inference from the trained VGAE: We show how, given a link prediction query specified by the user, the VGAE can be *dynamically* transformed to a conditional VAE (CVAE) for the query, without further training. A CVAE (Sohn, Lee, and Yan 2015) outputs a conditional ELBO that approximates the query probability: the conditional probability of one or more target links given the evidence specified in the query (cf. Figure 1).

Our contributions may be summarized as follows.

- A new query answering approach for ILP queries: estimate a conditional probability for a large class of ILP queries by inference from a single VGAE.
- A method for dynamically constructing a CVAE to answer a given graph query from the VGAE. A novel *sequential* CVAE design where node embeddings computed by the prior networks are used as initial embeddings for the recognition network.
- An application of our query answering method to new practically useful link prediction queries, including joint predictions of multiple links (e.g., a node neighborhood).

Paper Organization. We review related work, then the CVAE architecture and the conditional ELBO (Sohn, Lee, and Yan 2015). We describe inductive training of a modular VGAE based on the ELBO likelihood approximation. The space of link prediction queries is formally defined, with examples of previously investigated as well as novel queries. We show how the modules of the VGAE can be used to construct, for each ILP query, a CVAE.

Related Work

Inductive Graph Training. Inductive graph learning has been a major topic in recent research (Hamilton, Ying, and Leskovec 2017; Zeng et al. 2019; Rossi, Zhou, and Ahmed 2018). Our query answering approach can be used with any inductive encoder-decoder graph neural network architecture (e.g., the well-known GraphSage architecture (Hamilton, Ying, and Leskovec 2017)), which we train with the variational ELBO objective we describe below. To our knowledge this is the first inductive training with the variational ELBO objective.

CVAE for Graph Queries. Sohn, Lee, and Yan derived the conditional ELBO as a variational approximation for conditional probabilities and introduced the CVAE architecture for computing it. CVAEs are models for generating structured output, which makes them suitable for graph prediction. While they have been extensively applied to visual data, such as images and video, to our knowledge ours is the first application to graph queries.

Whereas in the original CVAE design the prior and recognition networks are separate from each other, for ILP we introduce a *novel sequential design*, where the training node embeddings computed by the prior network are passed to the recognition network as initial node embeddings. We show that different strategies for updating the initial node embeddings in the recognition network optimize different components of the conditional ELBO.

Inference from a Model. Previous work on graph neural networks has focused on specific predictive tasks specified by the researcher. To our knowledge, inference from a generative model to estimate conditional probabilities is a novel application of inductive neural graph learning. However, general query answering capabilities are standard in non-neural statistical-relational models, such as Markov Logic networks (Domingos and Lowd 2009) and Probabilistic Soft Logic (Huang et al. 2012). The fact that general query answering is a key capability of previous graph models motivates our goal of adding this capability to neural models. Statistical-relational models are based on very different assumptions and model classes (e.g. exponential random graph models) from VAEs, so we leave a direct comparison for future work.

Review of CVAE

A CVAE is a neural architecture for computing a **conditional** ELBO, denoted cELBO, which is a variational approximation to a conditional probability. Suppose we want to generate a distribution over outputs $\boldsymbol{Y}$ conditional on evidence $\boldsymbol{E}$. Essentially a CVAE follows the standard VAE setup for generating $\boldsymbol{Y}$, but conditions both the reconstruction probability and the prior latent distribution on $\boldsymbol{E}$. The cELBO approximation to the conditional log-likelihood is thus given by:

$$\ln(p(\boldsymbol{Y}|\boldsymbol{E})) \geq E_{\boldsymbol{z}\sim q(\boldsymbol{Z}|\boldsymbol{E},\boldsymbol{Y})}\big[\ln p(\boldsymbol{Y}|\boldsymbol{z},\boldsymbol{E})\big] \\ -KLD\big(q(\boldsymbol{Z}|\boldsymbol{E},\boldsymbol{Y})||p(\boldsymbol{Z}|\boldsymbol{E})\big) \quad (1)$$

The term $p(\boldsymbol{Z}|\boldsymbol{E})$ is the (conditional) prior distribution over the latent variables given the evidence $\boldsymbol{E}$. Usually this distribution is difficult to evaluate exactly. A CVAE approximates the (conditional) prior with a **prior network** $p(\boldsymbol{Z}|\boldsymbol{E}) \approx p_\psi(\boldsymbol{Z}|\boldsymbol{E})$. Similarly a CVAE uses a **recognition network** q_φ to approximate the posterior $q(\boldsymbol{Z}|\boldsymbol{E},\boldsymbol{Y})$.

In graph queries the target $\boldsymbol{Y}$ and evidence $\boldsymbol{E}$ specify parts of a graph (i.e., links and/or node features). Our main insight is that in a large class of graph queries the evidence $\boldsymbol{E}$ and the conjunction $\boldsymbol{E},\boldsymbol{Y}$ each describe a complete subgraph (possibly over different node sets). If a graph generative network model is trained inductively, *we can apply the same model both as prior network to compute* $p(\boldsymbol{Z}|\boldsymbol{E})$ *and as recognition network to compute* $q(\boldsymbol{Z}|\boldsymbol{E},\boldsymbol{Y})$. The following sections describe the details of this approach.

Variational Graph Auto-Encoder Architecture

Our approach can be applied with any encoder-decoder graph neural network (GNN) that outputs node representations and is trained with the variational ELBO objective (Kipf and Welling 2016; Hamilton 2020). Our modification is to build the encoder out of two modules: an attribute encoder that takes as input node attributes only, and a link encoder that outputs node representations given an adjacency matrix and initial node representations.

Data Format

An attributed graph is a pair $G = (V, E)$ comprising a finite set of nodes and edges where each node is assigned an k-dimensional attribute $\boldsymbol{x}_i$ with $k > 0$. An attributed graph can be represented by an $N \times N$ adjacency matrix $\boldsymbol{A}$ with $\{0, 1\}$ Boolean entries, together with an $N \times k$ node feature matrix $\boldsymbol{X}$.

Encoder-Decoder Architecture

Let $\boldsymbol{Z}$ be an $N \times d$ matrix that represents latent node embeddings. Like other VGAE models, the decoder $p_\alpha(\boldsymbol{A}|\boldsymbol{Z})$ generates links independently given node representations, as illustrated in Figure 2.

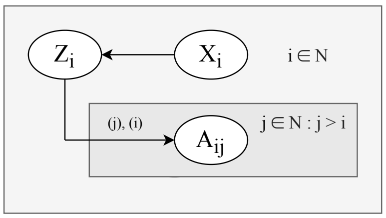


Figure 2: Generative diagram for our VGAE model

The graph encoder $q_\varphi(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{A})$ comprises an **attribute encoder** $p_\phi(\boldsymbol{Z}_0|\boldsymbol{X})$ and a **link encoder** $p_\theta(\boldsymbol{Z}|\boldsymbol{A}, \boldsymbol{Z}_0)$. The attribute encoder transforms node attributes to node representations. It can be implemented using any standard (non-relational) VAE architecture. The link encoder is applied to initial representations $\boldsymbol{Z}_0$ to compute updated node representations. It can be implemented with a message-passing graph neural network, such as GraphSage (Hamilton, Ying, and Leskovec 2017). We require that the encoder and decoder are inductive, meaning that they can output node embeddings for graphs of different sizes.

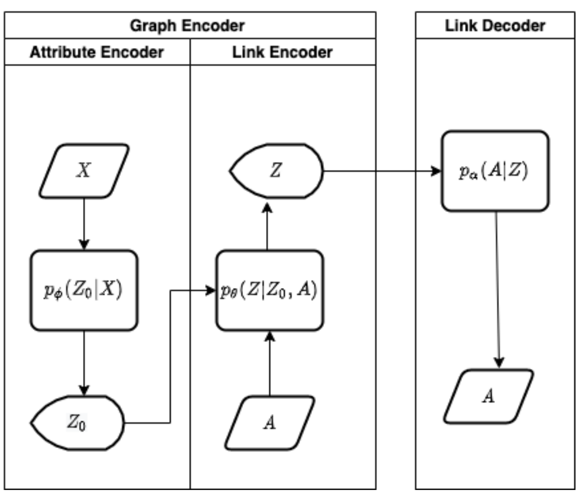


Figure 3: Generative Graph encoder-decoder Architecture for End-to-End Training

Training

In end-to-end training, the attribute encoder produces initial node representations from node attributes, and the link encoder final node representations from the initial ones; see Figure 3.

Motivation. Having an attribute encoder as a separate module from the link encoder provides more flexibility for leveraging node attributes, for example to handle queries that do not specify attributes for some nodes, or at the other extreme that require predicting a link from attribute information only.

Having a separate link encoder that takes as input initial node representations facilitates describing and implementing different methods for initializing node representations. For example below we describe how node representations computed by a CVAE prior network can be used to initialize a CVAE recognition network.

The approximate posterior distribution $q_{\phi,\theta}(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{A})$ is defined by the following generative process.

$$\boldsymbol{Z}_0 \sim p_\phi(\boldsymbol{Z}_0|\boldsymbol{X})$$
$$\boldsymbol{Z} \sim p_\theta(\boldsymbol{Z}|\boldsymbol{A}, \boldsymbol{Z}_0)$$

The training loss is the negative variational ELBO for link reconstruction (Kipf and Welling 2016):

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

Graph Queries

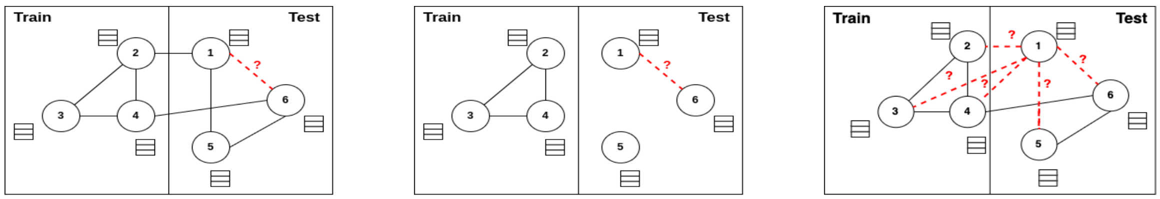
Answering a graph query requires assigning a joint probability to joint values for a set of *target* relational random variables $\boldsymbol{Y}$, given values of *evidence* random variables $\boldsymbol{E}$. A relational random variable corresponds to either a node attribute or an adjacency. Accordingly we define $\mathcal{X} = \{\boldsymbol{x}_u[i] : u \in V, 1 \leq i \leq k\}$ as the set of **attribute variables** and $\mathcal{A} = \{\boldsymbol{A}[u, v] : u \in V, v \in V\}$ as the set of **link variables**. A **link prediction query** is of the form:

$$\mathbb{P}(\boldsymbol{Y} = \boldsymbol{y}|\boldsymbol{E} = \boldsymbol{e}) \text{ where the } \textbf{target } \boldsymbol{Y} \subseteq \mathcal{A},$$
$$\text{the } \textbf{evidence } \boldsymbol{E} = (\boldsymbol{X}^{\boldsymbol{E}}, \boldsymbol{A}^{\boldsymbol{E}}) \text{ with } \boldsymbol{X}^{\boldsymbol{E}} \subseteq \mathcal{X}, \boldsymbol{A}^{\boldsymbol{E}} \subseteq \mathcal{A}$$

For query readability we group relational random variables according to the types of nodes that occur in them. The symbols U, U' etc. refer to a generic subset of nodes. The notation $\overline{U}$ denotes the complement of a node set $U \subseteq V$. The attribute variables of node type U are denoted as $\mathcal{X}_U \equiv \{\boldsymbol{x}_u[i] : u \in U, 1 \leq i \leq k\}$. The link variables that connect node type U_1 to node type U_2 are denoted as $\mathcal{A}_{U_1 \leftrightarrow U_2} \equiv \{\boldsymbol{A}[u, v] : u \in U_1, v \in U_2\}$. With some overload of notation, we write $X_U \subseteq \mathcal{X}_U$ for a generic set of attribute variables. Similarly $A_{U_1 \leftrightarrow U_2} \subseteq \mathcal{A}_{U_1 \leftrightarrow U_2}$ denotes a generic set of link variables, possibly empty. Table 1 provides examples of important graph queries. We distinguish the following query types.

Query	Name
$\mathbb{P}(\boldsymbol{A}[u,v] \mid \boldsymbol{X}, \mathcal{A} - \{(\boldsymbol{A}[u,v]\})$	Single Link Prediction from remaining graph
$\mathbb{P}(\boldsymbol{A}[u,v] \mid \mathcal{X}_{\{u,v\}})$	Single Link Prediction from attributes only
$\mathbb{P}(\mathcal{A}_{\{u\}\leftrightarrow V} \mid \mathcal{X}, \mathcal{A}_{\overline{\{u\}}\leftrightarrow V})$	Predict Node Neighbourhood
$\mathbb{P}(\mathcal{A}_{\overline{U}\leftrightarrow\overline{U}}, \mathcal{X}_{\overline{U}} \mid \mathcal{X}_U, \mathcal{A}_{U\leftrightarrow U}, A_{U\leftrightarrow\overline{U}})$	Graph Completion

Table 1: Examples of Graph Prediction Tasks Formulated as Relational Queries



(a) Single Link Prediction from remaining graph (b) Single Link Prediction from attributes only (c) Predict Node Neighbourhood of a single node

Figure 4: Illustrations of Inductive 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.
- A query is **attribute-complete** if for each node i, the evidence or the target contains either all or no attribute of i. (I.e., for each i, and attributes j_1, j_2, if $\boldsymbol{x}_i[j_1]$ appears in $\boldsymbol{E}$ resp. $\boldsymbol{Y}$, then $\boldsymbol{x}_i[j_2]$ appears in $\boldsymbol{E}$ resp. $\boldsymbol{Y}$.)

Link prediction queries are typically attribute-complete (cf. Table 1), and node classification queries are not. In the next section we describe a general procedure for constructing a CVAE for any attribute-complete link prediction query given an inductively trained VGAE, *without* further training. CVAEs can be used to answer both inductive queries and purely transductive link prediction queries, where all nodes are included in the training data. While our theory is general, our evaluation focuses on inductive link prediction to address the current research challenges. Figure 4 illustrates examples of important ILP queries.

CVAE Construction for Graph Queries

The cELBO for a graph query is given by

$$\ln(p(\boldsymbol{Y}|\boldsymbol{E})) \geq$$
$$E_{\boldsymbol{z}\sim q(\boldsymbol{Z}|\boldsymbol{E},\boldsymbol{Y})}\left[\ln p(\boldsymbol{Y}|\boldsymbol{z})\right] - KLD\big(q(\boldsymbol{Z}|\boldsymbol{E},\boldsymbol{Y})||p(\boldsymbol{Z}|\boldsymbol{E})\big) \quad (2)$$

where $\boldsymbol{Z}_{M\times d}$ represents a matrix of embeddings for all M nodes that appear in the query (both target and evidence). Equation (2) follows from Equation (1) together with the fact that target variables are independent of evidence given a node representations $\boldsymbol{z}$, which is implied by the generative model of Figure 2. The cELBO computed by a CVAE architecture was designed for "fill-in" tasks that aim to complete a partially specified structure (Sohn, Lee, and Yan 2015; Doersch 2016). Sohn et al. showed how the cELBO approximation can be used to estimate a joint distribution over an image completion given a partial image; a new contribution of our work is to evaluate it for *graph* completion.

CVAE Architecture

We now discuss how to compute the cELBO with a recognition network and a prior network. We observe that for an attribute-complete graph query, the evidence $\boldsymbol{E}$ and the conjunction $\boldsymbol{E}, \boldsymbol{Y}$ each describe an attribute-complete subgraph (given a closed-world assumption that unspecified links are absent). Since the VGAE uses an inductive encoder-decoder, *we can apply it both as prior network to compute* $p(\boldsymbol{Z}|\boldsymbol{E})$ *and as recognition network to compute* $q(\boldsymbol{Z}|\boldsymbol{E}, \boldsymbol{Y})$. The next section contains the precise equations. An important issue in inductive graph learning is whether the embeddings of observed training nodes should be updated given new test nodes (Hamilton, Ying, and Leskovec 2017). In our CVAE setting, we consider three approaches.

Separate The prior and recognition networks compute embeddings separately.

Sequential Evidence node embeddings from the prior network are used as initialization in the recognition network, which updates both evidence and test node embeddings.

Transfer Evidence node embeddings from the prior network are transferred to the recognition network, which updates only target node embeddings.

In the separate design, the prior network is used only to compute the KLD term of the cELBO. In the sequential and transfer designs, additionally the output of the prior network affects the output of the recognition network. Figure 5 illustrates the sequential design, the most complex option.

The cELBO (2) clarifies the trade-offs associated with the different update methods:

1. The transfer method has the least KLD term (namely 0) and also, we expect, the least reconstruction likelihood since it constrains the recognition network the most.
2. Separating the prior and recognition networks constrains the recognition network the least. We expect the largest reconstruction likelihood and the largest KLD term.

3. The sequential method is intermediate: Initialization regularizes evidence node embeddings from the recognition network to be similar to those from the prior network. We expect the reconstruction likelihood and the KLD term to lie between those of the separate and transfer methods.

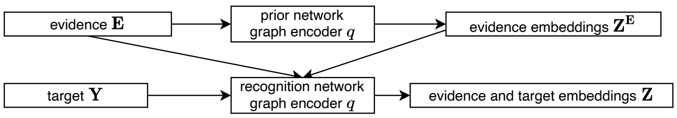


Figure 5: Sequential Design of the Conditional VAE. The VGAE encoder is used for both the prior and the recognition networks.

CVAE Implementation

Equations for the Prior Network Let $\boldsymbol{E} = (\boldsymbol{X}, \boldsymbol{A})$ be the evidence to be encoded. The modular design of the generative VGAE encoder makes it easy to extend it to attribute-complete evidence as follows: (1) If the node attributes $\boldsymbol{X}_i$ are specified in the evidence, apply the attribute encoder to compute $\boldsymbol{Z}_{0,i}$, the initial embedding of node i. (2) If the evidence specifies no attribute information for node i, use the VAE prior $p(\boldsymbol{Z})$ as initial embedding. (3) Apply the link encoder to the initial embeddings. are as follows. Let $\boldsymbol{Z}_{0,i}$ denote an initial embedding of node i; the node embeddings are stacked in a matrix $\boldsymbol{Z}_0$.

$$\boldsymbol{Z}_{0,i} \sim \begin{cases} p_\phi(\boldsymbol{Z}_{0,i}|\boldsymbol{X}_i) \text{ if } \boldsymbol{X}_i \subseteq \boldsymbol{X} \\ p(\boldsymbol{Z}) \text{ otherwise} \end{cases}$$
$$p(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{A}) = p_\theta(\boldsymbol{Z}|\boldsymbol{A}, \boldsymbol{Z}_0) \tag{3}$$

Equations for the Recognition Network

We write $q(\boldsymbol{Z}|\boldsymbol{Y}, \boldsymbol{E})$ for the distribution that represents the output of the recognition network. In the *separate* design, we run the encoder (3) on the joint $(\boldsymbol{Y}, \boldsymbol{E})$ information (missing node attributes are handled as in the prior network):

$$q(\boldsymbol{Z}|\boldsymbol{Y}, \boldsymbol{E}) = q_\varphi(\boldsymbol{Z}|\boldsymbol{Y}, \boldsymbol{E}) \text{ separate design}$$

In the transfer/sequential design, the initial embeddings for the recognition network are the evidence embeddings:

$$\boldsymbol{Z}_p \sim p(\boldsymbol{Z}|\boldsymbol{E}).$$

In link prediction queries, only the evidence contains attribute information (i.e., $\boldsymbol{X}^{\boldsymbol{Y}}$ is empty), which was already encoded by the attribute encoder in the prior network. The link encoder is therefore sufficient to update all embeddings for *sequential recognition*.

$$q(\boldsymbol{Z}|\boldsymbol{Y}, \boldsymbol{E}) = p_\theta(\boldsymbol{Z}|\boldsymbol{A}, \boldsymbol{Z}_p) \text{ sequential design}$$

where $\boldsymbol{A} = (\boldsymbol{A}^{\boldsymbol{Y}}, \boldsymbol{A}^{\boldsymbol{E}})$ is the union of all links specified in the target and evidence jointly.

Transfer recognition updates only the target node embeddings.

$$q(\boldsymbol{Z}^{\boldsymbol{Y}}|\boldsymbol{Y}, \boldsymbol{E}) = p_\theta(\boldsymbol{Z}^{\boldsymbol{Y}}|\boldsymbol{A}, \boldsymbol{Z}_p)$$
$$q(\boldsymbol{Z}^{\boldsymbol{E}}|\boldsymbol{Y}, \boldsymbol{E}) = p(\boldsymbol{Z}^{\boldsymbol{E}}|\boldsymbol{E}) \text{ transfer design}$$

Here the query node embeddings are partitioned as $\boldsymbol{Z} = (\boldsymbol{Z}^{\boldsymbol{E}}, \boldsymbol{Z}^{\boldsymbol{Y}})$ where $\boldsymbol{Z}^{\boldsymbol{E}}$ comprises embeddings for nodes mentioned in the query evidence, and $\boldsymbol{Z}^{\boldsymbol{Y}}$ the remaining nodes (mentioned in the target but not in the query).

Experimental Design

We plan to perform the following experiments in future work:

Inductive Generative Graph Learning We will evaluate different settings of the GraphSage architecture (Hamilton, Ying, and Leskovec 2017) for inductively training a VGAE, especially different neighborhood aggregation operators. Metrics include the training data ELBO as well as the downstream performance on ILP query answering. Supporting a large space of ILP queries allows us to examine the strengths and weaknesses of different inductive graph learning methods.

Inductive Link Prediction Our experiments will evaluate different methods for updating prior network embeddings, which is the main design choice in our CVGAE construction. Several recent papers provide strong baselines for ILP with single target links (Hao et al. 2020; Teru, Denis, and Hamilton 2020; Zhang and Chen 2019). We can directly compare our inference from a single model to the predictive accuracy of these customized solutions. For novel ILP queries that target multiple links, we can evaluate our CVAE approach by the likelihood it assigns to ground truth on test links. One of the advantages of variational inference is that it supports inference through sampling (e.g. find the most likely neighbourhood of a new node through sampling subgraphs). We will evaluate how well variational sampling supports multiple link prediction.

Conclusion

Graph query answering allows users to pose a wide range of link prediction queries through inference from a single model. Our work in progress describes a neural approach to graph query answering. After training a VGAE inductively, we can dynamically construct a CVAE for a given user query, which approximates the conditional query probability. A major design issue for inductive learning is how to update training node embeddings when making test node predictions (Hamilton, Ying, and Leskovec 2017). The CVAE framework provides a new perspective on this issue: Different update strategies correspond to different ways of connecting the prior and recognition networks, and to optimizing different components of the conditional ELBO. Inductive graph query answering has the potential to be an easy-to-use baseline with competitive accuracy for many practically useful tasks.

References

Doersch, C. 2016. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Domingos, P.; and Lowd, D. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan and Claypool Publishers.

Hamilton, W. L. 2020. *Graph representation learning*. 3. Morgan & Claypool Publishers.

Hamilton, W. L.; Ying, R.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 1025–1035.

Hao, Y.; Cao, X.; Fang, Y.; Xie, X.; and Wang, S. 2020. Inductive link prediction for nodes having only attribute information. *arXiv preprint arXiv:2007.08053*.

Huang, B.; Kimmig, A.; Getoor, L.; and Golbeck, J. 2012. Probabilistic soft logic for trust analysis in social networks. In *International Workshop on Statistical Relational Artificial Intelligence (StaRAI 2012)*.

Kipf, T. N.; and Welling, M. 2016. Variational graph autoencoders. *arXiv preprint arXiv:1611.07308*.

O'Bray, L.; Horn, M.; Rieck, B.; and Borgwardt, K. 2021. Evaluation Metrics for Graph Generative Models: Problems, Pitfalls, and Practical Solutions. *arXiv preprint arXiv:2106.01098*.

Rossi, R. A.; Zhou, R.; and Ahmed, N. K. 2018. Deep inductive graph representation learning. *IEEE Transactions on Knowledge and Data Engineering*, 32(3): 438–452.

Russell, S. 2015. Unifying logic and probability. *Communications of the ACM*, 58(7): 88–97.

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28: 3483–3491.

Teru, K.; Denis, E.; and Hamilton, W. 2020. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, 9448–9457. PMLR.

Zeng, H.; Zhou, H.; Srivastava, A.; Kannan, R.; and Prasanna, V. 2019. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*.

Zhang, M.; and Chen, Y. 2019. Inductive matrix completion based on graph neural networks. *arXiv preprint arXiv:1904.12058*.