Variational Flow Graphical Model

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper introduces a novel approach to embed flow-based models with hierarchical structures. The proposed model learns latent representation of high dimensional data via a message-passing scheme by carefully integrating normalizing flows in variational graphs. Meanwhile, the model can generate data representations with reduced latent dimensions, thus overcoming the drawbacks of many flow-based models, usually requiring a high dimensional latent space involving many trivial variables. With aggregation nodes, the model provides a convenient approach for data integration and graphical inference. Theoretical analysis and numerical experiments on synthetic and real datasets show the benefits and broad potentials of our proposed method.

1 Introduction

Graphical models [23, 11] are potent tools to combine the particular structure of a graph and probabilistic modeling, which provides a probabilistic (and hierarchical) characterization of variables. Due to their flexibility and ability to effectively learn and perform inference in large networks [19], they have attracted lots of interest. They have been applied in many fields, *e.g.* speech recognition [3], Quick Medical Reference (QMR) model [29] and energy-based model [12]. The quantity of interest in such models is the marginal distribution of the observed data, also known as the incomplete likelihood, noted $p(\mathbf{x})$. Most statistical learning tasks involve a parameterized model and their training procedure involves computing the maximum likelihood estimate defined as $\theta^* := \arg\max_{\theta \in \mathbb{R}^d} p_\theta(\mathbf{x})$. The maximization of such likelihood $p_\theta(\mathbf{x})$ in a parameterized model is closely related to the inference of the density $p_\theta(\mathbf{x}|\mathbf{z})$, as a subroutine during the training procedure. Note that in the above, $\mathbf{z}$ is the latent variable, and $p(\mathbf{x}, \mathbf{z})$ is the joint distribution of the complete data comprised of the observations x and z. Graphical models [23, 11] are potent tools to combine the particular structure of a graph and probabilistic modeling, which provides a probabilistic (and hierarchical) characterization of variables.

There are two general approaches for graphical inference: *exact inference* and *approximate inference*. Exact inference [28, 14] resorts to an exact numerical calculation procedure of the quantity of interest. However, in most cases, exactly inferring from $p_\theta(\mathbf{x}|\mathbf{z})$ is either *computationally involved* or simply *intractable*. Variational Inference (VI) is computationally efficient and has been applied to tackle the large scale inference problem [13, 10, 18, 21]. In Variational Inference, mean-field approximation [33] and variational message passing [4, 32] are two common approaches for graphical models. Those methods leverage families of simple and tractable distributions to approximate the intractable posterior $p(\mathbf{z}|\mathbf{x})$. However, such approximation is limited by the choice of distributions that are inherently unable to recover the true posterior, often leading to a loose lower bound. They also often lack a flexible structure to learn the intrinsic disentangled latent representation.

Contributions. Dealing with high dimensional data using graphical models exacerbates this systemic inability to model the latent structure of the data efficiently. To overcome these significant limitations, we propose a new framework, a variational hierarchical graphical flow model:

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

- **Hierarchical and Flow-Based:** Introducing the VARIATIONAL FLOW GRAPHICAL (VFG) model, we propose a novel graph architecture borrowing ideas from the *hierarchical latent data* modeling and *normalizing flow* concept to uncover the underlying complex structure of high dimensional data without any posterior sampling steps required in existing traditional variational models.
- **Information Aggregation:** Aggregation nodes are introduced to integrate hierarchical information through a forward-backward message passing scheme. The outcome of such design is a richer and tractable posterior distribution used as an approximation of the true posterior of the hidden node states in the structure. Model's interpretability can be improved by the proposed hierarchical aggregation scheme.
- **Numerical Inference:** We highlight the benefits of our VFG model on the applications to graph missing entries imputation problem and numerical inference on graphical datasets. We also show the potential application of VFG to tractable probabilistic inference on datasets. Algorithms have been developed to improve the training efficiency and node inference accuracy.
- **Representation Learning:** We specifically demonstrate that our model achieves to disentangle the factors of variation underlying the high dimensional data given as input.

Section 2 presents important concepts such as normalizing flows, and VI. Section 3 introduces the Variational Flow Graphical Model (VFG) model to tackle the latent relational structure learning of high dimensional data. Section 4 gives the algorithms to train VFG models. Section 5 discusses how to perform inference with a trained VFG model. Section 6 showcases the advantages of VFG on various tasks: missing values imputation on both synthetic and real datasets, and disentanglement learning. The Appendix is devoted to proofs and further analysis.

Notation: We denote by $[L]$ the set $\{1, \cdots, L\}$, for all $L > 1$, and by $\mathbf{KL}(p||q) := \int_{\mathcal{Z}} p(z) \log(p(z)/q(z)) \mathrm{d}z$ the Kullback-Leibler divergence from q to p, two probability density functions defined on the set $\mathcal{Z} \subset \mathbb{R}^d$ for any dimension $d > 0$.

2 Preliminaries

In this section, we first introduce the general principles and notations of normalizing flows and variational inference. Then, we explain how they can naturally be embedded with graphical models.

Variational Inference: Following the setting discussed above, the functional mapping $\mathbf{f}$: $\mathbf{x} \to \mathbf{z}$ can be viewed as an encoding process and the mapping $\mathbf{f}^{-1}$: $\mathbf{z} \to \mathbf{x}$ as a decoding one: $\mathbf{z} \sim p(\mathbf{z}), \mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{z})$. To learn the parameters θ, we maximize the following marginal log-likelihood $\log p_\theta(\mathbf{x}) = \log \int p(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z}$. Direct optimization of the log-likelihood is usually not an option due to the intractable latent structure. Instead VI employs a parameterized family of so-called variational distributions $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate the true posterior $p_\theta(\mathbf{z}|\mathbf{x}) \propto p(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z})$. The goal of VI is to minimize the distance, in terms of Kullback-Leibler ($\mathbf{KL}$) divergence, between the variational candidate and the true posterior $\mathbf{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$. This optimization problem can be shown to be equivalent to maximizing the following evidence lower bound (ELBO) objective, noted $\mathcal{L}(\mathbf{x}; \theta)$:

$$\log p_\theta(\mathbf{x}) \geqslant \mathcal{L}(\mathbf{x}; \theta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x}|\mathbf{z})\big] - \mathbf{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) := -\mathcal{F}(\theta, \phi)\,.$$

In Variational Auto-Encoders (VAEs, [18]), the calculation of the reconstruction term requires sampling from the posterior distribution along with using the reparameterization trick, i.e.,

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\big[\log p_\theta(\mathbf{x}|\mathbf{z})\big] \simeq \frac{1}{M} \sum_{m=1}^{M} \log p(\mathbf{x}|\mathbf{z}_m). \tag{1}$$

Here M is the sample number of latent variable from the posterior $q_\phi(\mathbf{z}|\mathbf{x})$ regarding data $\mathbf{x}$.

Normalizing Flows: Normalizing flows [16, 25] is a transformation of a simple probability distribution into a more complex distribution by a sequence of invertible and differentiable mappings, noted $\mathbf{f} : \mathcal{Z} \to \mathcal{X}$ between two random variables $z \in \mathcal{Z}$ of density $p(\mathbf{z})$ and $x \in \mathcal{X}$. Firstly introduced by [31] for single maps, it has been popularized in [7, 27] with deep neural networks for variational inference [25]. Flow-based models [7, 6, 5, 9, 24] are attractive approaches for density estimation as

they result in better performance enjoying the exact inference capability at a *low computational cost*. The observed variable $\mathbf{x} \sim p_\theta(\mathbf{x})$ is assumed to be distributed according to an unknown distribution $p_\theta(\mathbf{x})$ parameterized by a user-designed model θ. We focus on a finite sequence of transformations $\mathbf{f} := \mathbf{f}_1 \circ \mathbf{f}_2 \circ \cdots \circ \mathbf{f}_L$ such that, $\mathbf{x} = \mathbf{f}(\mathbf{z})\,, \quad \mathbf{z} = \mathbf{f}^{-1}(\mathbf{x}) \quad$ and $\quad \mathbf{z} \underset{\mathbf{f}_1^{-1}}{\overset{\mathbf{f}_1}{\rightleftharpoons}} \mathbf{h}^1 \underset{\mathbf{f}_2^{-1}}{\overset{\mathbf{f}_2}{\rightleftharpoons}} \mathbf{h}^2 \cdots \underset{\mathbf{f}_L^{-1}}{\overset{\mathbf{f}_L}{\rightleftharpoons}} \mathbf{x}$. By defining the aforementioned invertible maps $\{\mathbf{f}_\ell\}_{\ell=1}^L$, and by the chain rule and inverse function theorem, the variable $\mathbf{x} = \mathbf{f}(\mathbf{z})$ has a tractable probability density function (pdf) given as:

$$\log p_\theta(\mathbf{x}) = \log p(\mathbf{z}) + \log \left| \det\left(\frac{\partial \mathbf{z}}{\partial \mathbf{x}}\right) \right| = \log p(\mathbf{z}) + \sum_{i=1}^{L} \log \left| \det\left(\frac{\partial \mathbf{h}^i}{\partial \mathbf{h}^{i-1}}\right) \right|, \tag{2}$$

where we have $\mathbf{h}^0 = \mathbf{x}$ and $\mathbf{h}^L = \mathbf{z}$ for conciseness. The scalar value $\log |\det(\partial \mathbf{h}^i / \partial \mathbf{h}^{i-1})|$ is the logarithm of the absolute value of the determinant of the Jacobian matrix $\partial \mathbf{h}^i / \partial \mathbf{h}^{i-1}$, also called the log-determinant. Eq. (2) yields a simple mechanism to build families of distributions that, from an initial density and a succession of invertible transformations, returns tractable density functions that one can sample from (by sampling from the initial density and applying the transformations). [25] propose an approach to construct flexible posteriors by transforming a simple base posterior with a sequence of flows. Firstly a stochastic latent variable is draw from base posterior $\mathcal{N}(\mathbf{z}_0|\mu(\mathbf{x}), \sigma(\mathbf{x}))$. With K flows, latent variable $\mathbf{z}_0$ is transformed to $\mathbf{z}_k$.The reformed negative EBLO is given by

$$\begin{aligned} \mathcal{F}(\theta, \phi) =& \mathbb{E}_{q_\phi}\big[\log q_\phi(\mathbf{z}|\mathbf{x}) - \log p_\theta(\mathbf{x}, \mathbf{z})\big] \\ =& \mathbb{E}_{q_0}\big[\log q_0(\mathbf{z}_0|\mathbf{x}) - \log p_\theta(\mathbf{x}, \mathbf{z})\big] - \mathbb{E}_{q_0}\big[\sum_{k=1}^{K} \log \big| \det\left(\frac{\partial \mathbf{f}_k(\mathbf{z}_k; \psi_k)}{\partial \mathbf{z}_k}\right)\big|\big]. \end{aligned} \tag{3}$$

Here $\mathbf{f}_k$ is the kth flow with parameter ψ_k, i.e. $\mathbf{z}_K = \mathbf{f}_K \circ \cdots \mathbf{f}_2 \circ \mathbf{f}_1(\mathbf{z}_0)$. The parameters of the flows are considered as functions of data sample $\mathbf{x}$, and they determine the final distribution in amortized inference . In this paper we propose a framework that generalizes normalizing flow models [25, 2] to graphical variable inference.

3 Variational Flow Graphical Model

Assume that each data sample in a dataset has a sequence of sections or components and that there exist a relation between each of those sections and their corresponding latent variable. Then, it is possible to define a graphical model using normalizing flows, as introduced Section 2, leading to exact latent variable inference and log-likelihood evaluation of the model as well as relation discovery among data sections. We call this model a *Variational Flow Graphical Model* (VFG) and introduce it in the sequel.

3.1 Evidence Lower Bound of Variational Flow Graphical Models

A VFG model $\mathcal{G} = \{\mathcal{V}, \mathbf{f}\}$ consists of a set of nodes ($\mathcal{V}$) and a set of edges ($\mathbf{f}$). Figure 1-Right gives an illustration of a tree structure induced by a VFG model. An edge can be either a flow function or an identity function. There are two types of nodes in a VFG: *aggregation* nodes and *non-aggregation* nodes. A non-aggregation node connects other nodes with a single flow function or an identity function. An aggregation node has multiple children, and it connects with each of them with an identity function. In the following sections of this paper, identity function is considered as a special case of flow functions. We apply variational inference to assemble the layers of a VFG and learn parameters from data samples. Different from VAEs [18, 26], the recognition model (encoder) and the generative model (decoder) in a VFG share the same neural net structure and parameters. Moreover, the latent variables in a VFG lie in a hierarchy structure and are generated with deterministic flow functions.

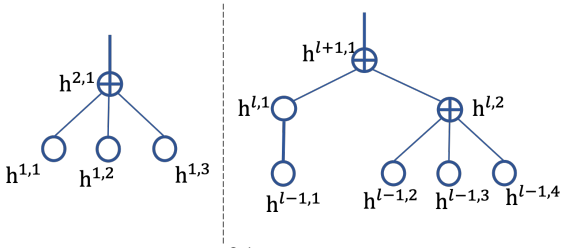


Figure 1: (Left) Node $\mathbf{h}^{2,1}$ connects its children with invertible functions. Messages from the children are aggregated at the parent, $\mathbf{h}^{2,1}$; $\oplus$ is an aggregation node, and circles stand for non-aggregation nodes.(Right) An illustration of the latent structure from layer $l-1$ to $l+1$. Thin lines are identity functions, and thick lines are flow functions.

The hierarchical generative network comprises L layers, $\mathbf{h}^l$ denotes the latent variable in layer l, and θ is the vector of model parameters. We use $\mathbf{h}^{l,i}$ to denote the ith node's latent variable in layer l, and $\mathbf{h}^{(j)}$ to represent node j's latent variable without specification of the layer number, and j is the node index on a tree or graph. The joint distribution of the hierarchical model is given by:

$$p_\theta(\mathbf{x}, \mathbf{h}) = p(\mathbf{h}^L)p(\mathbf{h}^{L-1}|\mathbf{h}^L) \cdots p(\mathbf{h}^1|\mathbf{h}^2)p(\mathbf{x}|\mathbf{h}^1)\,.$$

where $\mathbf{h} = \{\mathbf{h}^1, \cdots, \mathbf{h}^L\}$ denotes the set of latent variables of the model. The hierarchical generative model is given by factorization $p(\mathbf{x}|\mathbf{h}^L) = \mathbf{\Pi}_{l=1}^{L-1} p(\mathbf{h}^l|\mathbf{h}^{l+1})p(\mathbf{x}|\mathbf{h}^1)$, and the prior distribution is $p(\mathbf{h}^L)$.

The probability density function $p(\mathbf{h}^{l-1}|\mathbf{h}^l)$ in the generative model is parameterized with one or multiple invertible normalizing flow functions. We follow the variational inference to approximate the posterior distribution of latent variables. The hierarchical posterior (recognition model) is factorized as

$$q_\theta(\mathbf{h}|\mathbf{x}) = q(\mathbf{h}^1|\mathbf{x})q(\mathbf{h}^2|\mathbf{h}^1) \cdots q(\mathbf{h}^L|\mathbf{h}^{L-1}). \tag{4}$$

Evaluation of the posterior(recognition model) equation 4 involves forward information flows from the bottom of the tree to the top, and similarly, sampling the generative model takes the reverse direction. By leveraging the hierarchical conditional independence in both generative model and posterior, the ELBO regarding the model is given by

$$\log p_\theta(\mathbf{x}) \geqslant \mathcal{L}(\mathbf{x}; \theta) = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h}^{1:L})\big] - \sum_{l=1}^{L} \mathbf{KL}^l. \tag{5}$$

Here $\mathbf{KL}^l$ is the Kullback-Leibler divergence between the posterior and generative model in layer l. The first term in (5) evaluates data reconstruction. When $1 \leqslant l \leqslant L$,

$$\mathbf{KL}^l = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\mathbf{h}^{l+1})\big]. \tag{6}$$

When $l = L$, $\mathbf{KL}^L = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^L|\mathbf{h}^{L-1}) - \log p(\mathbf{h}^L)\big]$. It is easy to extend the computation of the ELBO (5) to DAGs with topology ordering of the nodes (and thus of the layers). Let $ch(i)$ and $pa(i)$ denote node i's child set and parent set, respectively. Then, the ELBO for a DAG structure reads:

$$\mathcal{L}(\mathbf{x}; \theta) = \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h})\big] - \sum_{i \in \mathcal{V} \setminus \mathcal{R}_\mathcal{G}} \mathbf{KL}^{(i)} - \sum_{i \in \mathcal{R}_\mathcal{G}} \mathbf{KL}\big(q(\mathbf{h}^{(i)}|\mathbf{h}^{ch(i)})||p(\mathbf{h}^{(i)})\big). \tag{7}$$

Here $\mathbf{KL}^{(i)} = \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\big[\log q(\mathbf{h}^{(i)}|\mathbf{h}^{ch(i)}) - \log p(\mathbf{h}^{(i)}|\mathbf{h}^{pa(i)})\big]$. $\mathcal{R}_\mathcal{G}$ is the set of root or source nodes of DAG $\mathcal{G} = \{\mathcal{V}, \mathbf{f}\}$. Assuming there are k leaf nodes on a tree or a DAG model, corresponding to k sections of the input sample $\mathbf{x} = [\mathbf{x}^{(1)}, ..., \mathbf{x}^{(k)}]$, then the hidden variables in both (5) and (7) are computed with forward and backward message passing. Next, we provide more details about the evaluation of ELBO.

3.2 ELBO Calculation

Maximizing the ELBO (5) equals to optimizing the parameters of the flows, θ. Similar to VAEs, we apply forward message passing (encoding) to approximate the posterior distribution of each layer's latent variables, and backward message passing (decoding) to generate the reconstructions as shown in Figure 2.

For the following sections, we use $\mathbf{h}^l$ to represent a sample from the posterior of layer l, and $\widehat{\mathbf{h}}^l$ for its reconstruction from the generative model. The calculation of the data reconstruction term in equation 5 requires samples of both the posterior and the generative model in each hidden layer. It corresponds to the encoding and decoding procedures

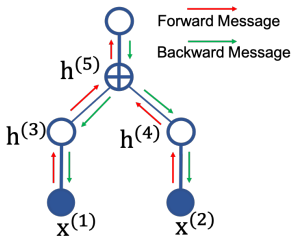


Figure 2: Forward and backward message passing to generate each node's hidden variable. Forward message passing approximates the posterior distribution of latent variables, and backward message passing generates the reconstructions.

in VAE model [18] as given by equation 1. Specifically, we have

$$\mathbf{h}^l \sim q(\cdot|\mathbf{h}^{l-1}) \qquad \text{where} \quad 1 \leqslant l \leqslant L\,., \tag{8}$$

$$\widehat{\mathbf{h}}^l \sim p(\cdot|\widehat{\mathbf{h}}^{l+1}) \qquad \text{where} \quad 0 \leqslant l \leqslant L-1\,. \tag{9}$$

At the root node, we have $\widehat{\mathbf{h}}^L = \mathbf{h}^L \sim q(\cdot|\mathbf{h}^{L-1})$. The computation of $\mathbf{h}^l$s and $\widehat{\mathbf{h}}^l$s with (8) and (9) requires message passing via the flow functions shown in Figure 2. A VFG is a deterministic model, samples in both equation 8 and equation 9 are directly generative with flow functions.

The reconstruction term in ELBO (5) can be computed with the backward message from the generative model $p(\mathbf{x}|\widehat{\mathbf{h}}^1)$, i.e.,

$$\begin{aligned}&\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h}^{1:L})\big] = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\widehat{\mathbf{h}}^{1:L})\big]\\ &\simeq \frac{1}{M}\sum_{m=1}^{M}\log p(\mathbf{x}|\widehat{\mathbf{h}}^{1:L}_m) = \frac{1}{M}\sum_{m=1}^{M}\log p(\mathbf{x}|\widehat{\mathbf{h}}^{1}_m) \simeq \log p(\mathbf{x}|\widehat{\mathbf{h}}^1).\end{aligned}$$

For a VFG model, we set $M = 1$. In the last term, $p(\mathbf{x}|\widehat{\mathbf{h}}^1)$ is either Gaussian or Binary distribution parameterized with $\widehat{\mathbf{x}}$ generated via the flow function with $\widehat{\mathbf{h}}^1$ as the input. For any $l \in [L]$, the calculation of the $\mathbf{KL}^l$ term is done in a similar manner, and equation 6 requires both forward message from the posterior and backward message from the generative model, i.e.,

$$\mathbf{KL}^l = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1})\big] \simeq \log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1}). \tag{10}$$

We use Laplace distribution to model latent distributions. $q(\cdot|\mathbf{h}^{l-1})$ is a Laplace with location and scale equal to the median and scale of a batch of $\mathbf{h}^l$, respectively; $p(\cdot|\widehat{\mathbf{h}}^{l+1})$ is a Laplace parameterized with $(\widehat{\mathbf{h}}^l, 1.0)$ as the location and scale parameters. Hence with $\mathbf{h}^l$ we can compute the log-likelihoods on RHS of (10) and thus the $\mathbf{KL}^l$ value.

In a tree structured VFG, each layer may consist of multiple nodes. In layer l, each node's $\mathbf{KL}$ value can be computed individually, thus $\mathbf{KL}^l = \sum_{i\in l}\mathbf{KL}^{(i)}$, and here each i represents a node. For a DAG structure, we can individually evaluate each node's $\mathbf{KL}$ term.

3.3 Aggregation Nodes

There are two approaches to aggregate signals from different nodes: average-based and concatenation-based. We rather focus on average-based aggregation in this paper, and Figure 3-Left gives an example denoted by the operator $\oplus$. Let $\mathbf{f}_{(i,j)}$ be the direct edge (function) from node i to node j, and $\mathbf{f}^{-1}_{(i,j)}$ or $\mathbf{f}_{(j,i)}$ defined as its inverse function. Then, we observe that at node i

$$\mathbf{h}^{(i)} = \frac{1}{|ch(i)|}\sum_{j\in ch(i)}\mathbf{f}_{(j,i)}(\mathbf{h}^{(j)}),$$

$$\widehat{\mathbf{h}}^{(i)} = \frac{1}{|pa(i)|}\sum_{j\in pa(i)}\mathbf{f}^{-1}_{(i,j)}(\widehat{\mathbf{h}}^{(j)})\,.$$

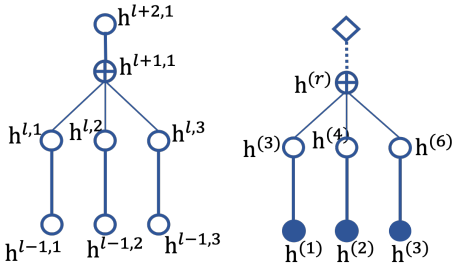


Figure 3: (Left) Aggregation node $\mathbf{h}^{l+1,1}$ has three children, $\mathbf{h}^{l,1}$, $\mathbf{h}^{l,2}$, and $\mathbf{h}^{l,3}$. (Right) A VFG model with one aggregation node, $\mathbf{h}^{(r)}$. Solid circles are nodes with observed values, and the diamond is the prior for the root node.

Notice that the above two equations hold even when node i has only one child or parent. In the sequel, we consider that all latent variables, noted $\mathbf{h}^{l,i}$, for all $l \in [L]$ and $i \in \mathbb{N}$, are distributed according to Laplace distributions. With the identity function between the parent and its children, there are two aggregation rules regarding an average aggregation node: (a) the parent value is the mean of its children, i.e., $\mathbf{h}^{(i)} = \frac{1}{|ch(i)|}\sum_{j\in ch(i)}\mathbf{h}^{(j)}$; (b) the child node have the same reconstruction value with its parent, i.e., $\widehat{\mathbf{h}}^{(j)} = \widehat{\mathbf{h}}^{(i)}, \forall j \in ch(i)$.

We use Figure 3-Right as an example to illustrate a simple model that has only one average aggregation node. Then (5) yields the ELBO,

$$\log p(\mathbf{x}) \geqslant \mathcal{L}(\mathbf{x};\theta) = \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\big[\log p(\mathbf{x}|\widehat{\mathbf{h}}^1)\big] - \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\big[\log q(\mathbf{h}^1|\mathbf{x}) - \log p(\mathbf{h}^1|\widehat{\mathbf{h}}^2)\big] - \mathbf{KL}\big(q(\mathbf{h}^2|\mathbf{h}^1)|p(\mathbf{h}^2)\big). \tag{11}$$

From Figure 3-Right, $\mathbf{h}^{(r)}$ is the root, and it has k children, $\mathbf{h}^{(t)}, t = 1, ..., k$, and $k = 3$. With $\mathbf{f}_t$ as the flow function connecting $\mathbf{h}^{(t)}$ and $\mathbf{x}^{(t)}$, according to the aggregation rules, we get:

$$\mathbf{h}^{(t)} = \mathbf{f}_t(\mathbf{x}^{(t)}), \quad \widehat{\mathbf{h}}^{(r)} = \mathbf{h}^{(r)} = \frac{1}{k}\sum_{t=1}^{k}\mathbf{h}^{(t)}, \quad \widehat{\mathbf{h}}^{(t)} = \widehat{\mathbf{h}}^{(r)},\ t = 1, ..., k\,. \tag{12}$$

Assume the data to be normally distributed, then

$$\log p(\mathbf{x}|\widehat{\mathbf{h}}^1) = -\sum_{t=1}^{k}\Big\{\underbrace{\frac{1}{2\sigma_{\mathbf{x}}^2}\big|\big|\mathbf{x}^{(t)} - \mathbf{f}_t^{-1}(\widehat{\mathbf{h}}^{(r)})\big|\big|_2^2}_{\text{By } \widehat{\mathbf{x}}^{(t)}=\mathbf{f}_t^{-1}(\widehat{\mathbf{h}}^{(t)})=\mathbf{f}_t^{-1}(\widehat{\mathbf{h}}^{(r)})}\Big\} + C,\ \log p(\mathbf{h}^1|\widehat{\mathbf{h}}^2) = -\sum_{t=1}^{k}\{\ \underbrace{\big|\big|\mathbf{f}_t(\mathbf{x}^{(t)}) - \widehat{\mathbf{h}}^{(r)}\big|\big|_1}_{\text{By } \widehat{\mathbf{h}}^2=\widehat{\mathbf{h}}^{(r)},\ \mathbf{h}^{(t)}=\mathbf{f}_t(\mathbf{x}^{(t)})}\ \} + C.$$

Here $\sigma_{\mathbf{x}}$ is a constant standard deviation. We notice that maximizing the ELBO, or minimizing the $\mathbf{KL}$ term of an aggregation node will force the model to satisfy aggregation rule (b).

4 Algorithms and Implementation

In this section, we develop the training algorithm, see Algorithm 1, that outputs the fitted vector of parameters resulting from the maximization of the ELBO objective function (5) or (7) depending on what graph structure is used. In Algorithm 1, the inference of the latent variables is performed via forwarding message passing, cf. Line 6, and their reconstructions are computed in backward message passing, cf. Line 11.

Unlike Variational Autoencoders (VAE), the variance of latent variables in a VFG is approximated rather than parameterized with neural networks. A VFG is a deterministic network passing latent variable values between nodes. The reconstruction (likelihood) terms in each layer are computed with forward and backward node states. Ignoring explicit neural network parameterized variances for all latent nodes enables us to use flow-based models as both the encoders and decoders. We thus obtain a deterministic ELBO objective (5)- (7) that can efficiently be optimized with standard stochastic optimizers.

Algorithm 1 Inference model parameters with forward and backward message propagation

1: **Input:** Data distribution $\mathcal{D}$, $\mathcal{G} = \{\mathcal{V}, \mathbf{f}\}$
2: **for** $s = 0, 1, ...$ **do**
3: Sample minibatch b samples $\{\mathbf{x}_1, ..., \mathbf{x}_b\}$ from $\mathcal{D}$;
4: **for** $i \in \mathcal{V}$ **do**
5: // forward message passing
6: $\mathbf{h}^{(i)} = \frac{1}{|ch(i)|}\sum_{j\in ch(i)}\mathbf{f}_{(j,i)}(\mathbf{h}^{(j)})$;
7: **end for**
8: $\widehat{\mathbf{h}}^{(i)} = \mathbf{h}^{(i)}$ if $i \in \mathcal{R}_{\mathcal{G}}$ or $i \in$ layer L;
9: **for** $i \in \mathcal{V}$ **do**
10: // backward message passing
11: $\widehat{\mathbf{h}}^{(i)} = \frac{1}{|pa(i)|}\sum_{j\in pa(i)}\mathbf{f}^{-1}_{(i,j)}(\widehat{\mathbf{h}}^{(j)})$;
12: **end for**
13: $\mathbf{h} = \{\mathbf{h}^{(t)}\big|t \in \mathcal{V}\}, \widehat{\mathbf{h}} = \{\widehat{\mathbf{h}}^{(t)}\big|t \in \mathcal{V}\}$;
14: Approximate the $\mathbf{KL}$ terms in ELBO for each layer with b samples;
15: Updating VFG model $\mathcal{G}$ with gradient ascending: $\theta_{\mathbf{f}}^{(s+1)} = \theta_{\mathbf{f}}^{(s)} + \nabla_{\theta_{\mathbf{f}}}\frac{1}{b}\sum_{i=1}^{b}\mathcal{L}(\mathbf{x}_b;\theta_{\mathbf{f}}^{(s)})$.
16: **end for**

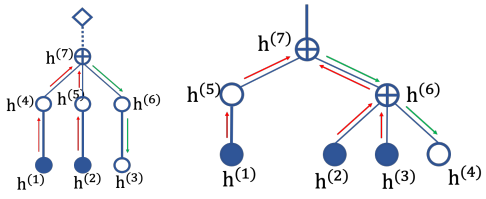


Figure 4: (Left) Inference on model with single aggregation node. Node 7 aggregates information from node 1 and 2, and pass down the updated state to node 3 for prediction. (Right) Inference on a tree model. Observed node states are gathered at node 7 to predict the state of node 4. Red and green lines are forward and backward messages, respectively.

4.1 Layer-wise Training

From a practical perspective, layer-wise training strategy can improve the efficiency of a model especially when it is constructed of more than two layers. In such a case, the parameters of only one layer are updated with backpropagation of the gradient of the loss function while keeping the other layers fixed at each optimization step. By maximizing the ELBO (5) or (7) with the above algorithm, the aggregation rules in Section 3.3 are expected to be satisfied. We can improve the inference on sub-graphs discussed in Section 5 by using the random masking method introduced in the sequel.

4.2 Random Masking

Inference on a VFG model requires the aggregation node's state to be imputed from observed children's state, as shown in (14). Then, unobserved children's state can be inferred from their parent. The inference ability of VFG via imputation can be reinforced by *masking out* some sections of the training samples. The training objective can be changed to force the model to impute the value of the masked sections. For example in a tree model, the alternative objective function reads

$$\mathcal{L}(\mathbf{x}, O_{\mathbf{x}}; \theta) = \sum_{t:1\leqslant t\leqslant k, t\notin O} \mathbb{E}_{q(\mathbf{h}^1|\mathbf{x}^{O_{\mathbf{x}}})}\Big[\log p(\mathbf{x}^{(t)}|\widehat{\mathbf{h}}^1)\Big] \tag{13}$$
$$- \sum_{l=1}^{L-1} \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\Big[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1})\Big] - \mathbf{KL}\big(q(\mathbf{h}^L|\mathbf{h}^{L-1})|p(\mathbf{h}^L)\big).$$

where $O_{\mathbf{x}}$ is the index set of leaf nodes with observation, and $\mathbf{x}^{O_{\mathbf{x}}}$ is the union of observed data sections. Considering a minibatch of training samples, the objectives function in (5) and (13) can thus be optimized sequentially. The training with *random masking* is described in Algorithm 2.

5 Inference on VFG Models

Given a trained VFG model, our goal in this subsection is to infer the state of any node given observed ones. Relations between variables at different nodes can also be inferred via our flow-based graphical model. The hidden state of the parent node j in a single aggregation model can be approximated by the observed children as follows:

$$\mathbf{h}^{(j)} = \frac{1}{|ch(j)\cap O|}\sum_{i\in ch(j)\cap O}\mathbf{h}^{(i)}, \tag{14}$$

where O is the set of observed leaf nodes, see Figure 4-left for an illustration. Observe that for either a tree or a DAG, the state of any given node is updated via messages received from its children. Figure 4 illustrates this inference mechanism for trees in which the structure enables us to perform message passing among the nodes. We derive the following Lemma establishing the relation between two leaf nodes:

Lemma 1. *Let $\mathcal{G}$ be a trained variational flow graphical model (with a tree structure) with L layers, and i and j are two leaf nodes with a as the closest common ancestor. Given observed value at node i, the value of node j can be approximated by $\widehat{\mathbf{x}}^j \approx \mathbf{f}_{(a,j)}(\mathbf{f}_{(i,a)}(\mathbf{x}^{(i)}))$. Here $\mathbf{f}_{(i,a)}$ is the flow function path from node i to node a.*

Algorithm 2 Inference model parameters with random masking

1: **Input:** Data distribution $\mathcal{D}$, $\mathcal{G} = \{\mathcal{V}, \mathbf{f}\}$
2: **for** $s = 0, 1, ...$ **do**
3: Sample minibatch b samples $\{\mathbf{x}_1, ..., \mathbf{x}_b\}$ from $\mathcal{D}$;
4: Optimize (5) with Line 4 to Line 15 in Algorithm 1;
5: Sample a subset of the k data sections as data observation set $O_{\mathbf{x}}$; $O \leftarrow O_{\mathbf{x}}$;
6: **for** $i \in \mathcal{V}$ **do**
7: // forward message passing
8: $\mathbf{h}^{(i)} = \frac{1}{|ch(i)\cap O|}\sum_{j\in ch(i)\cap O}\mathbf{f}_{(j,i)}(\mathbf{h}^{(j)})$;
9: $O \leftarrow O \cup \{i\}$ if $ch(i)\cap O \neq \emptyset$;
10: **end for**
11: $\widehat{\mathbf{h}}^{(i)} = \mathbf{h}^{(i)}$ if $i \in \mathcal{R}_{\mathcal{G}}$ or $i \in$ layer L;
12: **for** $i \in \mathcal{V}$ **do**
13: // backward message passing
14: $\widehat{\mathbf{h}}^{(i)} = \frac{1}{|pa(i)|}\sum_{j\in pa(i)}\mathbf{f}^{-1}_{(i,j)}(\widehat{\mathbf{h}}^{(j)})$;
15: **end for**
16: $\mathbf{h} = \{\mathbf{h}^{(t)}\big|t \in \mathcal{V}\cap O\}$, $\widehat{\mathbf{h}} = \{\widehat{\mathbf{h}}^{(t)}\big|t \in \mathcal{V}\}$;
17: Approximate the $\mathbf{KL}$ terms in ELBO for each layer with b samples;
18: Updating VFG with gradient of (13): $\theta_{\mathbf{f}}^{(s+1)} = \theta_{\mathbf{f}}^{(s)} + \nabla_{\theta_{\mathbf{f}}}\frac{1}{b}\sum_{i=1}^{b}\mathcal{L}(\mathbf{x}_b, O_{\mathbf{x}}; \theta_{\mathbf{f}}^{(s)})$,
19: **end for**

Considering the normalizing flow (2), we have the following identity for each node of the graph structure:

$$p(\mathbf{h}^{(i)}|\mathbf{h}^{pa(i)}) = p(\mathbf{h}^{pa(i)})\big|\det(\frac{\partial \mathbf{h}^{pa(i)}}{\partial \mathbf{h}^{(i)}})\big| = p(\mathbf{h}^{pa(i)})\big|\det(\mathbf{J}_{\mathbf{h}^{pa(i)}}(\mathbf{h}^{(i)}))\big|.$$

Remark 1. *Let O be the set of observed leaf nodes, j be an unobserved node, and a the closest ancestor of $O\cup\{a\}$. Then the state of j can be imputed with $\widehat{\mathbf{x}}^j \approx \mathbf{f}_{(a,j)}(\mathbf{f}_{(O,a)}(\mathbf{x}^{(i)}))$, where $\mathbf{f}_{(O,a)}$ is the flow function path from all nodes in O to a.*

6 Numerical Experiments

The first application we present is the imputation of missing values. We compare our method with several baseline models on a synthetic dataset. The second set of experiments is to evaluate VFG models on three different datasets, i.e., MNIST, Caltech101, and Omniglot, with ELBO and likelihoods as the score. The third application we present is the task of learning the disentangled latent representations that separate the explanatory factors of variations in the data, see [1]. For that latter application, the model is trained and evaluated on the MNIST handwritten digits dataset.

6.1 Evaluation on Inference with Missing Entries Imputation

We now focus on the task of imputing missing entries in a graph structure. For all the following experiments, the models are trained on the training set and are used to infer the missing entries of samples in the testing set. We use MSE regarding the prediction and ground truth as the imputation metric in the comparison of different methods.

Baseline Methods: We use the following baselines for data imputation:

- *Mean Value:* Using training set mean values to replace the missing entries in the testing set.
- *Iterative Imputation:* A strategy for imputing missing values by modeling each feature with missing values as a function of other features in a Round-Robin fashion.
- *KNN:* To use K-Nearest Neighbor for data imputation, we compare the non-missing entries of each sample to the training set and use the average of top k samples as imputation values
- *Multivariate Imputation by Chained Equation (MICE):* This method impute the missing entries with multiple rounds of inference. This method can handle different type of data.

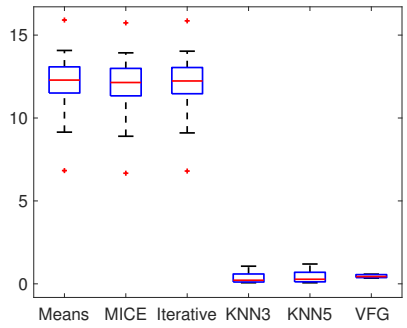


Figure 5: Synthetic datasets: MSE boxplots of VFG and baseline methods.

Synthetic dataset: In this set of experiments, we study the proposed model with synthetic datasets. We generate 10 synthetic datasets (using different seeds) of 1 300 data points, 1 000 for the training phase of the model, 300 for imputation testing. Each data sample has 8 dimensions with 2 latent variables. Let $z_1 \sim \mathcal{N}(0, 1.0^2)$ and $z_2 \sim \mathcal{N}(1.0, 2.0^2)$ be the latent variables. For a sample $\mathbf{x}$, we have $x_1 = x_2 = z_1, x_3 = x_4 = 2\sin(z_1), x_5 = x_6 = z_2$, and $x_7 = x_8 = z_2^2$. In the testing dataset, x_3, x_4, x_7, and x_8 are missing. We use a VFG model with a single average aggregation node that has four children, and each child connects the parent with a flow function consisting of 3 coupling layers [7]. Each child takes 2 variables as input data section, and the latent dimension of the VFG is 2. We compare, Figure 5, our VFG method with the baselines described above using boxplots on MSE errors for those 10 simulated datasets. We can see that the proposed VFG model performs much better than mean value, iterative, and MICE methods. VFG achieves similar MSE values but with much smaller performance variance compared with KNN methods.

6.2 ELBO and Likelihood

In Table 1, the negative evidence lower bound (-ELBO) and the estimated negative likelihood (NLL) for baseline methods on three datasets, MNIST, Caltech101, and Omniglot. The results of the baseline methods are from [2]. These methods are VAE based methods enhanced with normalizing flows. They use 16 flows to improve the posterior estimation. SNF is orthogonal sylvester flow method with a bottleneck of M = 32. We set the VFG coupling block number with $d = 4$, and following [2] we run multiple times to get the mean and standard derivation as well. Compared to VAE based methods,

the proposed VFG model can achieve significant improvement on both ELBO and NLL values on these datasets.

Model	MNIST		Caltech101		Omniglot	
	-ELBO	NLL	-ELBO	NLL	-ELBO	NLL
VAE [18]	86.55 ± 0.06	82.14 ± 0.07	110.80 ± 0.46	99.62 ± 0.74	104.28 ± 0.39	97.25 ± 0.23
Planer [25]	86.06 ± 0.31	81.91 ± 0.22	109.66 ± 0.42	98.53 ± 0.68	102.65 ± 0.42	96.04 ± 0.28
IAF [17]	84.20 ± 0.17	80.79 ± 0.12	111.58 ± 0.38	99.92 ± 0.30	102.41 ± 0.04	96.08 ± 0.16
SNF [2]	83.32 ± 0.06	80.22 ± 0.03	104.62 ± 0.29	93.82 ± 0.62	99.00 ± 0.04	93.77 ± 0.03
VFG	$\mathbf{80.80 \pm 0.76}$	$\mathbf{63.66 \pm 0.14}$	$\mathbf{67.26 \pm 0.53}$	$\mathbf{65.74 \pm 0.84}$	$\mathbf{80.16 \pm 0.73}$	$\mathbf{78.65 \pm 0.66}$

Table 1: Negative log-likelihood and free energy (negative evidence lower bound) for static MNIST, Caltech101, and Omniglot.

6.3 Latent Representation Learning on MNIST

In this set of experiments, we evaluate Variational Flow Graphical Models on latent representation learning of the MNIST dataset [20]. We construct a two layer VFG model, and set $\lambda = 1$. The first layer consists of one aggregation node with four children, and each child has an input dimension 14×14. The second layer is a single flow model. The latent dimension for this model is 196. Figure 7 presents the VFG tree structure used here. Following [30], the VFG model is trained with image labels to improve the disentanglement of the latent representation of the input data. We set the parameters of $\mathbf{h}^L$'s prior distribution as a function of image label, i.e., $\lambda^L(u)$, where u denotes the image label. In practice, we use 10 trainable λ^Ls regarding the 10 digits. The images in the second row of Figure 6 are reconstructions of MNIST samples extracted from the testing set, displayed in the first row of the same Figure, using our proposed VFG model.

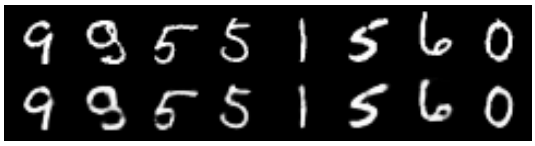


Figure 6: (Top row) original MNIST digits. (Bottom row) reconstructed images using VFG.

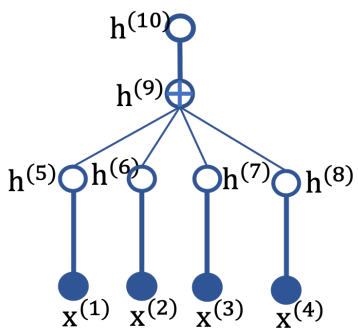


Figure 7: The tree structure for MNIST.

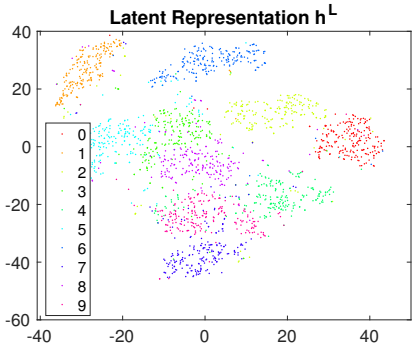


Figure 8: MNIST: t-SNE plot of latent variables from VFG learned with labels.

Figure 8 shows the t-distributed stochastic neighbor embedding (t-SNE) [22] plot of $2,000$ testing images's latent variables learned with our model, and 200 for each digit. We observe from Figure 8, that VFG can learn separated latent representations to distinguish different hand-written numbers.

7 Conclusion

In this paper, we propose VFG, a variational flow graphical model that aims at bridging the gap between normalizing flows and the paradigm of graphical models. Our VFG model learns the hierarchical latent structure of the input data through message passing between latent nodes. The posterior inference, of the latent nodes given input observations, is facilitated by the careful embedding of normalizing flow in the general graph structure. Experiments on different datasets illustrate the effectiveness of the model. Future work includes applying our VFG model to fine grained data relational structure learning and reasoning.

References

[1] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[2] R. v. d. Berg, L. Hasenclever, J. M. Tomczak, and M. Welling. Sylvester normalizing flows for variational inference. *arXiv preprint arXiv:1803.05649*, 2018.

[3] J. A. Bilmes and C. Bartels. Graphical model architectures for speech recognition. *IEEE signal processing magazine*, 22(5):89–100, 2005.

[4] C. M. Bishop, D. Spiegelhalter, and J. Winn. Vibes: A variational inference engine for bayesian networks. In *Advances in neural information processing systems*, pages 793–800, 2003.

[5] N. De Cao, W. Aziz, and I. Titov. Block neural autoregressive flow. In *Uncertainty in Artificial Intelligence*, pages 1263–1273. PMLR, 2020.

[6] L. Dinh, D. Krueger, and Y. Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[7] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real nvp. *ArXiv*, abs/1605.08803, 2016.

[8] B. Efron et al. Defining the curvature of a statistical problem (with applications to second order efficiency). *The Annals of Statistics*, 3(6):1189–1242, 1975.

[9] J. Ho, X. Chen, A. Srinivas, Y. Duan, and P. Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. *arXiv preprint arXiv:1902.00275*, 2019.

[10] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[11] E. R. Hruschka, E. R. Hruschka, and N. F. Ebecken. Bayesian networks for imputation in classification problems. *Journal of Intelligent Information Systems*, 29(3):231–252, 2007.

[12] M. I. Jordan, editor. *Learning in Graphical Models*. MIT Press, Cambridge, MA, USA, 1999.

[13] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[14] D. Kahle, T. Savitsky, S. Schnelle, and V. Cevher. Junction tree algorithm. *Stat*, 631, 2008.

[15] I. Khemakhem, D. Kingma, R. Monti, and A. Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. volume 108 of *Proceedings of Machine Learning Research*, pages 2207–2217, Online, 26–28 Aug 2020. PMLR.

[16] D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018.

[17] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improving variational inference with inverse autoregressive flow. *arXiv preprint arXiv:1606.04934*, 2016.

[18] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[19] D. Koller, N. Friedman, L. Getoor, and B. Taskar. Graphical models in a nutshell. *Introduction to statistical relational learning*, 43, 2007.

[20] Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010.

[21] Q. Liu and D. Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in neural information processing systems*, pages 2378–2386, 2016.

[22] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[23] D. Madigan, J. York, and D. Allard. Bayesian graphical models for discrete data. *International Statistical Review/Revue Internationale de Statistique*, pages 215–232, 1995.

[24] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

[25] D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

[26] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[27] O. Rippel and R. P. Adams. High-dimensional probability estimation with deep density models. *arXiv preprint arXiv:1302.5125*, 2013.

[28] S. Sanner and E. Abbasnejad. Symbolic variable elimination for discrete and continuous graphical models. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[29] M. Shwe, B. Middleton, D. Heckerman, M. Henrion, E. Horvitz, H. Lehmann, and G. Cooper. A probabilistic reformulation of the quick medical reference system. In *Proceedings of the Annual Symposium on Computer Application in Medical Care*, page 790. American Medical Informatics Association, 1990.

[30] P. Sorrenson, C. Rother, and U. Köthe. Disentanglement by nonlinear ica with general incompressible-flow networks (gin). In *Ninth International Conference on Learning Representations*, 2020.

[31] E. G. Tabak, E. Vanden-Eijnden, et al. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

[32] J. Winn and C. M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6(Apr):661–694, 2005.

[33] E. P. Xing, M. I. Jordan, and S. Russell. A generalized mean field algorithm for variational inference in exponential families. *arXiv preprint arXiv:1212.2512*, 2012.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Supplemental File for 'Variational Flow Graphical Model'

The proposed variational flow graphical models assemble flow functions with tree or DAG structures via variational inference on aggregation nodes.

A Additional Numerical Experiments

In all the experiments of the paper, we use the same coupling block [7] to construct different flow functions. The coupling block consists in three fully connected layers (of dimension 64) separated by two RELU layers along with the coupling trick. Each flow function has block number $b \geqslant 2$. All latent variables, $\mathbf{h}^i, i \in \mathcal{V}$ are forced to be non-negative via Sigmoid or RELU functions. Non-negativeness can help the model to identify sparse structures of the latent space.

A.1 California Housing Dataset

We further investigate the method on a real dataset. The California Housing dataset has 8 feature entries and $20\,640$ data samples. We use the first $20\,000$ samples for training and 100 of the rest for testing. We get 4 data sections, and each section contains 2 variables. In the testing set, the second section is assumed missing for illustration purposes, as the goal is to impute this missing section. Here, we construct a tree structure VFG with 2 layers. The first layer has two aggregation nodes, and each of them has two children. The second layer consists of one aggregation node that has two children connecting with the first layer. Each flow function has 4 coupling blocks. We can see Table 2 that our model yields significantly better results than any other method in terms of prediction error.

Methods	*Imputation MSE*
Mean Value	1.993
MICE	1.951
Iterative Imputation	1.966
KNN (k=3)	1.974
KNN (k=5)	1.969
VFG	**1.356**

Table 2: California Housing dataset: Imputation Mean Squared Error (MSE) results.

A.2 Inference on DAGs

In this set of experiments, we compare VFGs against Bayesian networks and sum-product networks on inference capabilities.

A.3 Representation Learning with MNIST

For MNIST, we construct a tree structure VFG model depicted in Figure 7. In the first layer, there are 4 flow functions, and each of them takes 14×14 image blocks as the input. Thus a 28×28 input image is divided into four 14×14 blocks as the input of VFG model. The four nodes are aggregated as the input of the upper layer flow.

A.3.1 Latent Representation Learning on MNIST

Figure 9 presents the t-SNE plot of the root latent variables from VFG trained without labels. The figure clearly shows that even without label information, different digits' representation are roughly scattered in different areas. Compared to Figure 8 in section 6.3, label information indeed can improve the latent representation learning.

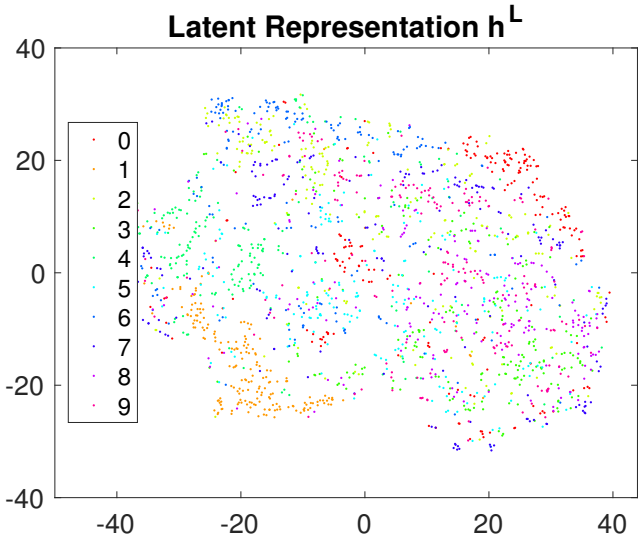
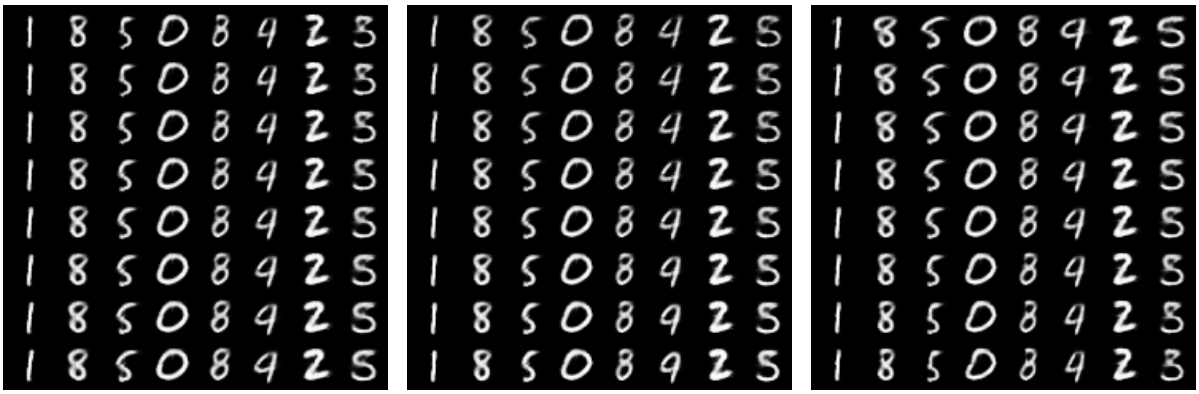


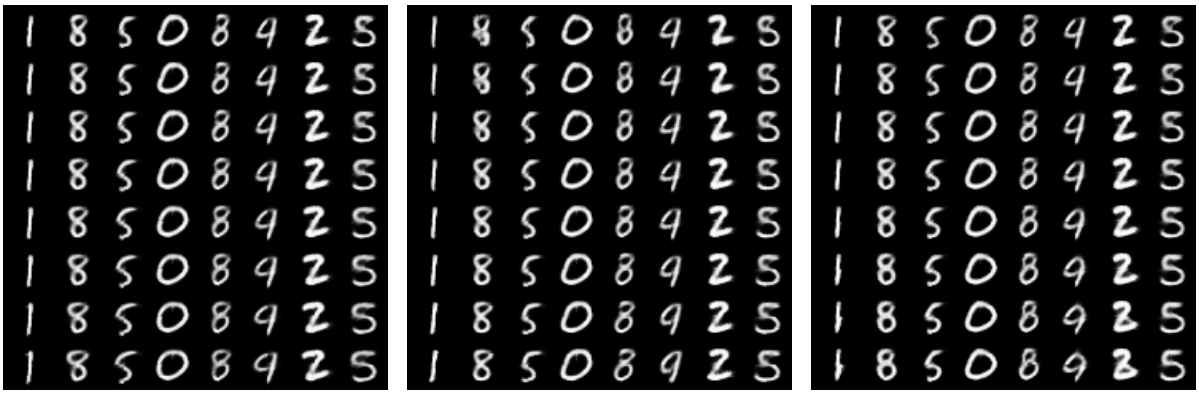
Figure 9: MNIST: t-SNE plot of latent variables from VFG learned without labels.

A.3.2 Disentanglement on MNIST

We study disentanglement on MNIST with our proposed VFG model introduced in section 6.3. But different from the model in section 6.3, here, the distribution parameter λ for all latent variables are set to be trainable across all layers. Each digit has its trainable vector, $\lambda \in \mathbb{R}^d$ that is used across all layers. To show the disentanglement of learned latent representation, we first obtain the root latent variables of a set of images through forward message passing. Each latent variable's values are changed increasingly within a range centered at the value of the latent variable obtained from last step. This perturbation is performed for each image in the set. Figure 10 shows the change of images by increasing one latent variable from a small value to a larger one. The figure presents some of the latent variables that have obvious effects on images, and most of the $d = 196$ variables do not impact the generation significantly. Latent variables $i = 6$ and $i = 60$ control the digit width. Variable $i = 19$ affects the brightness. $i = 92, i = 157$ and some of the variables not displayed here control the style of the generated digits.



(a) i=6, Width (b) i=19, Brightness (c) i=60, Upper width



(d) i=92 (e) i=119, Angle (f) i=157

Figure 10: MNIST: Increasing each latent variable from a small value to a larger one.

B ELBO Calculation

The recognition model in a VFG is the neural network (encoder) used to approximate the posterior of latent variables. With invertible neural networks (flows), the recognition model and the generative model in a VFG share the same structure and parameters. As shown in Figure 11, the recognition and generative models are realized with forward and backward message passing, respectively.

Maximize the ELBOs (5,7) requires evaluation of both the reconstruction and the **KL** terms. It involves samples from both the recognition and generative models. In this section, we first gives the conditional distributions in both generative model and the posterior, then give more details on ELBO computation. We start from tree models, and it is easy to extend to DAG models.

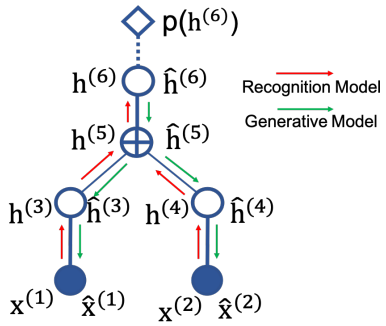


Figure 11: The recognition model consists of froward message from data to approximate the posterior distributions; the generative model is realized by backward message from the root node.

B.1 Distributions of Latent Variables

B.1.1 Generative Model

In a tree VFG, the sample reconstruction in the generative model consists of layer-wise backward message passing, i.e., latent variable generation in each layer. For any $l, 0 \leq l \leq L-1$, latent variable $\widehat{\mathbf{h}}^l$ is propagated from layer $l+1$ via the flow function $\mathbf{f}_l$ between the two layers with $\widehat{\mathbf{h}}^l = \mathbf{f}_l^{-1}(\widehat{\mathbf{h}}^{l+1})$.

The prior $p(\mathbf{h}^L)$ for the root latent variable $\mathbf{h}^L$ is Laplace(0,1). With a sample $\widehat{\mathbf{h}}^L$ from the posterior, i.e., $\widehat{\mathbf{h}}^L = \mathbf{h}^L \sim q(\cdot|\mathbf{h}^{L-1})$, the conditional distribution for latent variable in layer l is $p(\cdot|\widehat{\mathbf{h}}^{l+1}) :=$ Laplace$(\widehat{\mathbf{h}}^l, 1)$. Here the location parameter is generated from layer $l+1$, i.e., $\widehat{\mathbf{h}}^l = \mathbf{f}_l^{-1}(\widehat{\mathbf{h}}^{l+1})$.

For a latent variable $\mathbf{h}^l$ sampling from the posterior, its log-likelihood regarding $p(\cdot|\widehat{\mathbf{h}}^{l+1})$ in (10) is given by

$$\log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1}) = -\|\mathbf{h}^l - \widehat{\mathbf{h}}^l\|_1 - d \cdot \log 2.$$

Here $d = dim(\mathbf{h}^l)$. Hence, minimizing $\mathbf{KL}$s is to minimize the ℓ_1 distance between latent variables and their reconstructions.

B.1.2 Recognition Model

The forward message passing in the recognition model consists of layer-wise sample generation. In layer $l, 1 \leq l \leq L$, latent variable $\mathbf{h}^l$ is propagated from layer $l-1$ via the flow function $\mathbf{f}_{l-1}$ between the two layers with $\mathbf{h}^l = \mathbf{f}_{l-1}(\mathbf{h}^{l-1})$.

We assume each entry of hidden variable $\mathbf{h}^l$ follows a Laplace distribution, i.e., $\mathbf{h}^l_j \sim$ Laplace(μ^l_j, s^l_j) for layer l's jth entry. Here μ^l_j is the location and s^l_j is the scale. Compared with other distributions, Laplace can introduce sparsity to the model and it works well in practice. At level $l \in [L]$, we set $q(\cdot|\mathbf{h}^{l-1}) :=$ Laplace$(\mu^l, \mathbf{s}^l)$ with

$$\mu^l = \text{median}\big(H\big), \;\; \mathbf{s}^l = \frac{1}{B}\sum_{b=1}^{B} |\mathbf{h}^l(\mathbf{x}_b) - \mu^l| \,. \tag{15}$$

Here $H = \{\mathbf{h}^l(\mathbf{x}_b) | 1 \leqslant b \leqslant B\}$ is a batch of latent values generated from a batch of data samples with size B, i.e., $X_B = \{\mathbf{x}_b | 1 \leqslant b \leqslant B\}$. The median operation is performed element-wisely. For each $\mathbf{x}_b$, $\mathbf{h}^l(\mathbf{x}_b) = \mathbf{f}^{l-1}\big(\mathbf{h}^{l-1}(\mathbf{x}_b)\big)$.

B.2 KL Term

For any $l, 1 \leq l \leq L-1$, the calculation of the $\mathbf{KL}^l$ term (6) requires message passing and samples from both recognition and generative models, i.e.,

$$\mathbf{KL}^l = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1})\big] \simeq \log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1}). \tag{16}$$

Here $q(\cdot|\mathbf{h}^{l-1})$ is a Laplace with location and scale equal to the median and scale defined in equation 15; $p(\cdot|\widehat{\mathbf{h}}^{l+1})$ is a Laplace parameterized with $(\widehat{\mathbf{h}}^l, 1.0)$ as discussed in B.1.1. Hence with $\mathbf{h}^l$ we can compute the log-likelihoods on RHS of (16) and thus the $\mathbf{KL}^l$ value. When $l = L$, $\mathbf{KL}^L = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^L|\mathbf{h}^{L-1}) - \log p(\mathbf{h}^L)\big] \simeq \log q(\mathbf{h}^L|\mathbf{h}^{L-1}) - \log p(\mathbf{h}^L)$.

Assuming there are k leaf nodes on a tree or a DAG model, corresponding to k sections of the input sample $\mathbf{x} = [\mathbf{x}^{(1)}, ..., \mathbf{x}^{(k)}]$, then the hidden variables in both (5) and (7) are computed with forward and backward message passing. Next, we provide more details about the nodes.

In practice, we set $M = 1$ for efficiency. With a batch of training samples, $\mathbf{x}_b, 1 \leqslant b \leqslant B$, the structure of flow functions make the forward and backward message passing very efficient, and thus the estimation of the ELBO.

B.3 Reconstruction Term

The reconstruction term in ELBO (5) can be computed with the backward message from the generative model $p(\mathbf{x}|\widehat{\mathbf{h}}^1)$, i.e.,

$$\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h}^{1:L})\big] = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\widehat{\mathbf{h}}^{1:L})\big]$$
$$\simeq \frac{1}{M}\sum_{m=1}^{M}\log p(\mathbf{x}|\widehat{\mathbf{h}}_m^{1:L}) = \frac{1}{M}\sum_{m=1}^{M}\log p(\mathbf{x}|\widehat{\mathbf{h}}_m^{1}) \simeq \log p(\mathbf{x}|\widehat{\mathbf{h}}^1).$$

For a VFG model, we set $M = 1$. In the last term, $p(\mathbf{x}|\widehat{\mathbf{h}}^1)$ is either Gaussian or Binary distribution parameterized with $\widehat{\mathbf{x}}$ generated via the flow function with $\widehat{\mathbf{h}}^1$ as the input.

C Aggregation Node

Let $\mathbf{f}_{(i,j)}$ be the direct edge (function) from node i to node j, and $\mathbf{f}_{(i,j)}^{-1}$ or $\mathbf{f}_{(j,i)}$ defined as its inverse function. Then, at an aggregation node i that has multiple ($|ch(i)|$) children, its latent variable in forward message passing is the mean of all children's output, i.e.,

$$\mathbf{h}^{(i)} = \frac{1}{|ch(i)|}\sum_{j\in ch(i)}\mathbf{f}_{(j,i)}(\mathbf{h}^{(j)}).$$

On the other hand, if node i in a DAG has multiple parents, the reconstruction of its latent variable is the mean of all parents' output , i.e.,

$$\widehat{\mathbf{h}}^{(i)} = \frac{1}{|pa(i)|}\sum_{j\in pa(i)}\mathbf{f}_{(i,j)}^{-1}(\widehat{\mathbf{h}}^{(j)})\,.$$

Notice that the above two equations hold even when node i has only one child or parent.

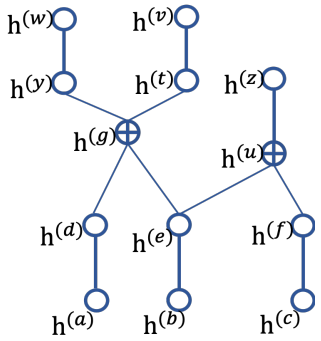


Figure 12: Aggregation nodes on a DAG.

Besides averaging, the aggregation nodes also ensure the latent variable are *consistent*. We use node i in the DAG presented in Figure 12 as an example. Node i has two parents, u and v; and two children, d and e.

The aggregation consistent means

D More Details on Inference

Lemma 1. *Let $\mathcal{G}$ be a well trained tree structured variational flow graphical model with L layers, and i and j are two leaf nodes with a as the closest common ancestor. Given observed value at node i, the value of node j can be approximated with $\widehat{\mathbf{x}}^{(j)} \approx \mathbf{f}_{(a,j)}(\mathbf{f}_{(i,a)}(\mathbf{x}^{(i)}))$. Here $\mathbf{f}_{(i,a)}$ is the flow function path from node i to node a.*

Proof. Without loss generality, we assume that there are relationships among different data sections, and the value of one section can be partially or approximately imputed by other sections. According to the aggregation rule (b) discussed in section 3.3, at an aggregation node a, the latent value of a child node j has the same reconstruction value as the parent node. The reconstruction of the child node j can be approximated with the reconstruction of the parent node, i.e., $\widehat{\mathbf{h}}^{(j)} \approx \mathbf{f}_{(a,j)}(\widehat{\mathbf{h}}^{a)})$. Recalling the reconstruction term in the ELBO (5), at each node we have $\mathbf{h}^{(a)} \approx \widehat{\mathbf{h}}^{(a)}$. Hence for node a's descendent j, we have $\widehat{\mathbf{h}}^{(j)} \approx \mathbf{f}_{(a,j)}(\mathbf{h}^{(a)})$, and $\mathbf{f}_{(a,j)}$ is the flow function path from a to j. The value of node a can be approximated by the value of its descendent i that has observation, i.e., $\mathbf{h}^{(a)} \approx \mathbf{f}_{(i,a)}(\mathbf{h}^{(i)})$. Hence, we have $\widehat{\mathbf{x}}^{(j)} \approx \mathbf{f}_{(a,j)}(\mathbf{f}_{(i,a)}(\mathbf{x}^{(i)}))$.

□

E Derivation of the ELBOs for Trees and DAGs

E.1 ELBO of Tree Models

Let each data sample has k sections, i.e., $\mathbf{x} = [\mathbf{x}^{(1)}, ..., \mathbf{x}^{(k)}]$. VFGs are graphical models that can integrate different sections or components of the dataset. We assume that for each pair of connected nodes, the edge is an invertible flow function. The vector of parameters for all the edges is denoted by θ. The forward message passing starts from $\mathbf{x}$ and ends at $\mathbf{h}^L$, and backward message passing in the reverse direction. We start with the hierarchical generative tree network structure illustrated by an example in Figure 13. Then the marginal likelihood term of the data reads

$$p(\mathbf{x}|\theta) = \sum_{\mathbf{h}^1,\ldots,\mathbf{h}^L} p(\mathbf{h}^L|\theta)p(\mathbf{h}^{L-1}|\mathbf{h}^L,\theta)\cdots p(\mathbf{x}|\mathbf{h}^1,\theta)\,.$$

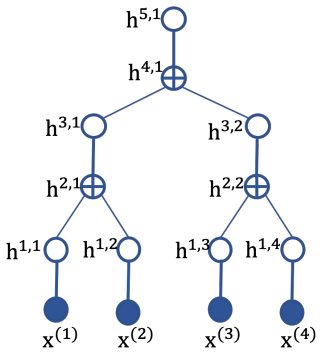


Figure 13: A tree VFG with $L = 5$ and three aggregation nodes.

The hierarchical prior distribution is given by factorization

$$p(\mathbf{h}) = p(\mathbf{h}^L)\mathbf{\Pi}_{l=1}^{L-1} p(\mathbf{h}^l|\mathbf{h}^{l+1}). \tag{17}$$

The probability density function $p(\mathbf{h}^{l-1}|\mathbf{h}^l)$ in the prior is modeled with one or multiple invertible normalizing flow functions. The hierarchical posterior (recognition network) is factorized as

$$q_\theta(\mathbf{h}|\mathbf{x}) = q(\mathbf{h}^1|\mathbf{x})q(\mathbf{h}^2|\mathbf{h}^1)\cdots q(\mathbf{h}^L|\mathbf{h}^{L-1}). \tag{18}$$

Draw samples from the prior (17) involves sequential conditional sampling from the top of the tree to the bottom, and computation of the posterior (18) takes the reverse direction. Notice that

$$q(\mathbf{h}|\mathbf{x}) = q(\mathbf{h}^1|\mathbf{x})q(\mathbf{h}^{2:L}|\mathbf{h}^1)\,.$$

With the hierarchical structure of a tree, we further have

$$q(\mathbf{h}^{l:L}|\mathbf{h}^{l-1}) = q(\mathbf{h}^l|\mathbf{h}^{l-1})q(\mathbf{h}^{l+1:L}|\mathbf{h}^l\mathbf{h}^{l-1}) = q(\mathbf{h}^l|\mathbf{h}^{l-1})q(\mathbf{h}^{l+1:L}|\mathbf{h}^l) \tag{19}$$

$$p(\mathbf{h}^{l:L}) = p(\mathbf{h}^l|\mathbf{h}^{l+1:L})p(\mathbf{h}^{l+1:L}) = p(\mathbf{h}^l|\mathbf{h}^{l+1})p(\mathbf{h}^{l+1:L}) \tag{20}$$

By leveraging the conditional independence in the chain structures of both posterior and prior, the derivation of trees' ELBO becomes easier.

$$\begin{aligned}\log p(\mathbf{x}) &= \log \int p(\mathbf{x}|\mathbf{h})p(\mathbf{h})d\mathbf{h} \\ &= \log \int \frac{q(\mathbf{h}|\mathbf{x})}{q(\mathbf{h}|\mathbf{x})}p(\mathbf{x}|\mathbf{h})p(\mathbf{h})d\mathbf{h} \\ &\geqslant \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h}) - \log q(\mathbf{h}|\mathbf{x}) + \log p(\mathbf{h})\big] = \mathcal{L}(x;\theta).\end{aligned}$$

The last step is due to the Jensen inequality. With $\mathbf{h} = \mathbf{h}^{1:L}$,

$$\begin{aligned}&\log p(\mathbf{x}) \geqslant \mathcal{L}(x;\theta) \\ =&\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h}^{1:L}) - \log q(\mathbf{h}^{1:L}|\mathbf{x}) + \log p(\mathbf{h}^{1:L})\big] \\ =&\underbrace{\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h}^{1:L})\big]}_{\text{(a) Reconstruction of the data}} - \underbrace{\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^{1:L}|\mathbf{x}) - \log p(\mathbf{h}^{1:L})\big]}_{\mathbf{KL}^{1:L}}\end{aligned} \tag{21}$$

With conditional independence in the hierarchical structure, we have

$$q(\mathbf{h}^{1:L}|\mathbf{x}) = q(\mathbf{h}^{2:L}|\mathbf{h}^1\mathbf{x})q(\mathbf{h}^1|\mathbf{x}) = q(\mathbf{h}^{2:L}|\mathbf{h}^1)q(\mathbf{h}^1|\mathbf{x}).$$

The second term of (21) can be further expanded as

$$\mathbf{KL}^{1:L} = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^1|\mathbf{x}) + \log q(\mathbf{h}^{2:L}|\mathbf{h}^1) - \log p(\mathbf{h}^1|\mathbf{h}^{2:L}) - \log p(\mathbf{h}^{2:L})\big].$$

Similarly, with conditional independence of the hierarchical latent variables, $p(\mathbf{h}^1|\mathbf{h}^{2:L}) = p(\mathbf{h}^1|\mathbf{h}^2)$. Thus

$$\begin{aligned}\mathbf{KL}^{1:L} =&\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^1|\mathbf{x}) - \log p(\mathbf{h}^1|\mathbf{h}^2) + \log q(\mathbf{h}^{2:L}|\mathbf{h}^1) - \log p(\mathbf{h}^{2:L})\big] \\ =&\underbrace{\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^1|\mathbf{x}) - \log p(\mathbf{h}^1|\mathbf{h}^2)\big]}_{\mathbf{KL}^1} + \underbrace{\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^{2:L}|\mathbf{h}^1) - \log p(\mathbf{h}^{2:L})\big]}_{\mathbf{KL}^{2:L}}.\end{aligned}$$

We can further expand the $\mathbf{KL}^{2:L}$ term following similar conditional independent rules regarding the tree structure. At level l, we get

$$\mathbf{KL}^{l:L} = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^{l:L}|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^{l:L})\big].$$

With (19) and (20), it is easy to show that

$$\mathbf{KL}^{l:L} = \underbrace{\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\mathbf{h}^{l+1})\big]}_{\mathbf{KL}^l} + \underbrace{\mathbb{E}_{q(\mathbf{h}^{l:L}|\mathbf{x})}\big[\log q(\mathbf{h}^{l+1:L}|\mathbf{h}^l) - \log p(\mathbf{h}^{l+1:L})\big]}_{\mathbf{KL}^{l+1:L}}. \tag{22}$$

The ELBO (21) can be written as

$$\mathcal{L}(\mathbf{x};\theta) = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h}^{1:L})\big] - \sum_{l=1}^{L-1}\mathbf{KL}^l - \mathbf{KL}^L. \tag{23}$$

When $1 \leqslant l \leqslant L-1$

$$\mathbf{KL}^l = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\mathbf{h}^{l+1})\big]. \tag{24}$$

As discussed in section B, evaluation of the terms in (23) requires samples of both the posterior and the prior in each layer of the tree structure. According to conditional independence, the expectation regarding variational distribution layer l just depends on layer $l-1$. We can simplify the expectation each term of (23) with the default assumption that all latent variables are generated regarding data sample $\mathbf{x}$. Therefore the ELBO (23) can be simplified as

$$\mathcal{L}(\mathbf{x};\theta) = \mathbb{E}_{q(\mathbf{h}^1|\mathbf{x})}\big[\log p(\mathbf{x}|\widehat{\mathbf{h}}^1)\big] - \sum_{l=1}^{L}\mathbf{KL}^l. \tag{25}$$

The $\mathbf{KL}$ term (24) becomes

$$\mathbf{KL}^l = \mathbb{E}_{q(\mathbf{h}^l|\mathbf{h}^{l-1})}\big[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) - \log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1})\big].$$

When $l = L$,

$$\mathbf{KL}^L = \mathbb{E}_{q(\mathbf{h}^L|\mathbf{h}^{L-1})}\big[\log q(\mathbf{h}^L|\mathbf{h}^{L-1}) - \log p(\mathbf{h}^L)\big].$$

E.2 Improve ELBO Estimation with Flows

To compute the EBLO, one way is to approximate $\mathbf{KL}$ terms with the latent values generated from a batch of training data samples. In this paper we follow the approach in [25, 17, 2] using normalizing flows to further improve posterior estimation. At each layer, minimizing $\mathbf{KL}$ term is to is to optimize the parameters of the network so that the posterior is closer to the prior. As shown in Figure 11, for layer l, we can take the encoding-decoding procedures (discussed in section B) as transformation of the posterior distribution from layer l to $l+1$, and then transform it back. By counting in the transformation difference [25, 17, 2], the $\mathbf{KL}$ at layer l becomes

$$\begin{aligned}\mathbf{KL}^l =& \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\left[\log q(\mathbf{h}^l|\mathbf{h}^{l-1}) + \log\left|\det\frac{\partial \mathbf{h}^l}{\partial \mathbf{h}^{l+1}}\right| + \log\left|\det\frac{\partial \widehat{\mathbf{h}}^{l+1}}{\partial \widehat{\mathbf{h}}^l}\right| - \log p(\mathbf{h}^l|\widehat{\mathbf{h}}^{l+1})\right] \\ \simeq& \frac{1}{M}\sum_{m=1}^{M}\left[\log q(\mathbf{h}_m^l|\mathbf{h}^{l-1}) + \log\left|\det\frac{\partial \mathbf{h}_m^l}{\partial \mathbf{h}_m^{l+1}}\right| + \log\left|\det\frac{\partial \widehat{\mathbf{h}}_m^{l+1}}{\partial \widehat{\mathbf{h}}_m^l}\right| - \log p(\mathbf{h}_m^l|\widehat{\mathbf{h}}_m^{l+1})\right].\end{aligned}$$

E.3 ELBO of DAG Models

Note that if we reverse the edge directions in a DAG, the resulting graph is still a DAG graph. The nodes can be listed in a topological order regarding the DAG structure as shown in Figure 14.

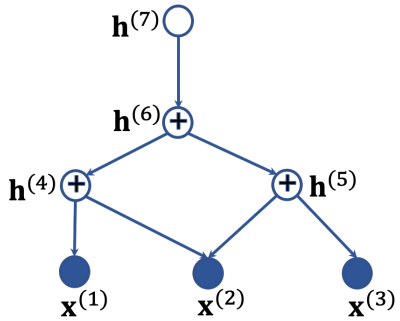


Figure 14: A DAG with inverse topology order $\{$ {1,2,3}, {4,5}, {6}, {7} $\}$, and they correspond to layers 0 to 3.

By taking the topology order as the layers in tree structures, we can derive the ELBO for DAG structures. Assume the DAG structure has L layers, and the root nodes are in layer L. We denote by $\mathbf{h}$ the vector of latent variables, then following (21) we develop the ELBO as

$$\begin{aligned}\log p(\mathbf{x}) \geqslant \mathcal{L}(x;\theta) =& \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\left[\log\frac{p(\mathbf{x},\mathbf{h})}{q(\mathbf{h}|\mathbf{x})}\right] \\ =& \underbrace{\mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{h})\right]}_{\text{Reconstruction of the data}} \underbrace{-\mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\left[\log q(\mathbf{h}|\mathbf{x}) - \log p(\mathbf{h})\right]}_{\mathbf{KL}}.\end{aligned} \tag{26}$$

Similarly the KL term can be expanded as in the tree structures. For nodes in layer l

$$\mathbf{KL}^{l:L} = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^{l:L}|\mathbf{h}^{1:l-1}) - \log p(\mathbf{h}^{l:L})\big].$$

Note that $ch(l)$ may include nodes from layers lower than $l-1$, and $pa(l)$ may include nodes from layers higher than l. Some nodes in l may not have parent. Based on conditional independence with the topology order of a DAG, we have

$$q(\mathbf{h}^{l:L}|\mathbf{h}^{1:l-1}) = q(\mathbf{h}^l|\mathbf{h}^{1:l-1})q(\mathbf{h}^{l+1:L}|\mathbf{h}^l) = q(\mathbf{h}^l|\mathbf{h}^{1:l-1})q(\mathbf{h}^{l+1:L}|\mathbf{h}^{1:l}) \tag{27}$$

$$p(\mathbf{h}^{l:L}) = p(\mathbf{h}^l|\mathbf{h}^{l+1:L})p(\mathbf{h}^{l+1:L}) \tag{28}$$

Following (22) and with (27-28), we have

$$\mathbf{KL}^{l:L} = \mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[\log q(\mathbf{h}^l|\mathbf{h}^{1:l-1}) - \log p(\mathbf{h}^l|\mathbf{h}^{l+1:L})\big] + \underbrace{\mathbb{E}_{q(\mathbf{h}^{l:L}|\mathbf{x})}\big[\log q(\mathbf{h}^{l+1:L}|\mathbf{h}^{1:l}) - \log p(\mathbf{h}^{l+1:L})\big]}_{\mathbf{KL}^{l+1:L}}.$$

Furthermore,

$$q(\mathbf{h}^l|\mathbf{h}^{1:l-1}) = q(\mathbf{h}^l|\mathbf{h}^{ch(l)}), \qquad p(\mathbf{h}^l|\mathbf{h}^{l+1:L}) = p(\mathbf{h}^l|\mathbf{h}^{pa(l)}).$$

Hence,

$$\mathbf{KL}^{l:L} = \underbrace{\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[q(\mathbf{h}^l|\mathbf{h}^{ch(l)}) - p(\mathbf{h}^l|\mathbf{h}^{pa(l)})\big]}_{\mathbf{KL}^l} + \mathbf{KL}^{l+1:L} \tag{29}$$

For nodes in layer l,

$$\mathbf{KL}^l = \sum_{i\in l} \underbrace{\mathbb{E}_{q(\mathbf{h}^{1:L}|\mathbf{x})}\big[q(\mathbf{h}^{(i)}|\mathbf{h}^{ch(i)}) - p(\mathbf{h}^{(i)}|\mathbf{h}^{pa(i)})\big]}_{\mathbf{KL}^{(i)}}.$$

Recurrently applying (29) to (26) yields

$$\mathcal{L}(\mathbf{x};\theta) = \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\big[\log p(\mathbf{x}|\mathbf{h})\big] - \sum_{i\in\mathcal{V}\setminus\mathcal{R}_\mathcal{G}} \mathbf{KL}^{(i)} - \sum_{i\in\mathcal{R}_\mathcal{G}} \mathbf{KL}\big(q(\mathbf{h}^{(i)}|\mathbf{h}^{ch(i)})||p(\mathbf{h}^{(i)})\big).$$

For node i,

$$\mathbf{KL}^{(i)} = \mathbb{E}_{q(\mathbf{h}|\mathbf{x})}\big[q(\mathbf{h}^{(i)}|\mathbf{h}^{ch(i)}) - p(\mathbf{h}^{(i)}|\mathbf{h}^{pa(i)})\big].$$

F Theoretical Justifications for Latent Representation Learning

The proposed Variational Flow Graphical models provide approaches to integrate multi-modal (multiple natures of data) or multi-source (collected from various sources) data. With invertible flow functions, we analyze the identifiability [15, 30] of the VFG in this section. We assume that each input data point has k sections, and denote by $\mathbf{h}^{(t)}$, the latent variable for section t, namely $\mathbf{x}^{(t)}$. Suppose the distribution of the latent variable $\mathbf{h}^{(t)}$, conditioned on $\mathbf{u}$, is a factorial member of the exponential family with $m > 0$ sufficient statistics, see [8] for more details on exponential families. Here $\mathbf{u}$ is an additional observed variable which can be considered as covariates. The general form of the exponential distribution can be expressed as

$$p_{\mathbf{h}^{(t)}}(\mathbf{h}^{(t)}|\mathbf{u}) = \Pi_{i=1}^d \frac{Q_i(h^{(t,i)})}{Z_i(\mathbf{u})} \exp\left[\sum_{j=1}^m T_{i,j}(h^{(t,i)})\lambda_{i,j}(\mathbf{u})\right], \tag{30}$$

where Q_i is the base measure, $Z_i(\mathbf{u})$ is the normalizing constant, $T_{i,j}$ are the component of the sufficient statistic and $\lambda_{i,j}$ the corresponding parameters, depending on the variable $\mathbf{u}$. Data section variable $\mathbf{x}^{(t)}$ is generated with some complex, invertible, and deterministic function from the latent space as in:

$$\mathbf{x}^{(t)} = \mathbf{f}_t^{-1}(\mathbf{h}^{(t)}, \epsilon), \tag{31}$$

where ϵ is some additional random noise in the generation of $\mathbf{x}^{(t)}$. Let $\mathbf{T} = [\mathbf{T}_1, ..., \mathbf{T}_d]$, and $\lambda = [\lambda_1, ..., \lambda_d]$. We define the domain of the inverse flow $\mathbf{f}_t^{-1}$ as $\mathcal{H} = \mathcal{H}_1 \times ... \times \mathcal{H}_d$. The parameter set $\widehat{\Theta} = \{\widehat{\theta} := (\widehat{\mathbf{T}}, \widehat{\lambda}, \mathbf{g})\}$ is defined in order to represent the model learned by a piratical algorithm. Let $\mathbf{z}^{(t)}$ be one sample's latent variable recovered by the algorithm regarding $\mathbf{h}^{(t)}$. In the limit of infinite data and algorithm convergence, we establish the following theoretical result regarding the identifiability of the sufficient statistics $\mathbf{T}$ in our model (30).

Theorem 1. *Assume that the observed data is distributed according to the model given by (30) and (31). Let the following assumptions holds,*

(a) The sufficient statistics $T_{ij}(h)$ are differentiable almost everywhere and their derivatives $\partial T_{i,j}/\partial_h$ are nonzero almost surely for all $h \in \mathcal{H}_i$, $1 \leqslant i \leqslant d$ and $1 \leqslant j \leqslant m$.

(b) There exist $(dm + 1)$ distinct conditions $\mathbf{u}^{(0)}$, ..., $\mathbf{u}^{(dm)}$ such that the matrix

$$\mathbf{L} = [\lambda(\mathbf{u}^{(1)}) - \lambda(\mathbf{u}^{(0)}), ..., \lambda(\mathbf{u}^{(dm)}) - \lambda(\mathbf{u}^{(0)})]$$

of size $dm \times dm$ is invertible.

Then the model parameters $\mathbf{T}(\mathbf{h}^{(t)}) = \mathbf{A}\widehat{\mathbf{T}}(\mathbf{z}^{(t)}) + \mathbf{c}$. Here $\mathbf{A}$ is a $dm \times dm$ invertible matrix and $\mathbf{c}$ is a vector of size dm.

Proof. The conditional probabilities of $p_{\mathbf{T},\lambda,\mathbf{f}_t^{-1}}\left(\mathbf{x}^{(t)}|\mathbf{u}\right)$ and $p_{\widehat{\mathbf{T}},\widehat{\lambda},\mathbf{g}}\left(\mathbf{x}^{(t)}|\mathbf{u}\right)$ are assumed to be the same in the limit of infinite data. By expanding the probability density functions with the correct change of variable, we have

$$\log p_{\mathbf{T},\lambda}(\mathbf{h}^{(t)}|\mathbf{u}) + \log\left|\det \mathbf{J}_{\mathbf{f}_t}(\mathbf{x}^{(t)})\right| = \log p_{\widehat{\mathbf{T}},\widehat{\lambda}}(\mathbf{z}^{(t)}|\mathbf{u}) + \log\left|\det \mathbf{J}_{g^{-1}}(\mathbf{x}^{(t)})\right|.$$

Let $\mathbf{u}^{(0)}, ..., \mathbf{u}^{(dm)}$ be from condition (b). We can subtract this expression of $\mathbf{u}^{(0)}$ from some $\mathbf{u}^{(v)}$. The Jacobian terms will be removed since they do not depend $\mathbf{u}$,

$$\log p_{\mathbf{h}^{(t)}}(\mathbf{h}^{(t)}|\mathbf{u}^{(v)}) - \log p_{\mathbf{h}^{(t)}}(\mathbf{h}^{(t)}|\mathbf{u}^{(0)}) = \log p_{\mathbf{z}^{(t)}}(\mathbf{z}^{(t)}|\mathbf{u}^{(v)}) - \log p_{\mathbf{z}^{(t)}}(\mathbf{z}^{(t)}|\mathbf{u}^{(0)}). \tag{32}$$

Both conditional distributions in equation 32 belong to the exponential family. Eq. (32) thus reads

$$\begin{aligned}
&\sum_{i=1}^{d}\left[\log\frac{Z_i(\mathbf{u}^{(0)})}{Z_i(\mathbf{u}^{(v)})} + \sum_{j=1}^{m} T_{i,j}(\mathbf{h}^{(t)})\left(\lambda_{i,j}(\mathbf{u}^{(v)}) - \lambda_{i,j}(\mathbf{u}^{(0)})\right)\right] \\
=&\sum_{i=1}^{d}\left[\log\frac{\widehat{Z}_i(\mathbf{u}^{(0)})}{\widehat{Z}_i(\mathbf{u}^{(v)})} + \sum_{j=1}^{m} \widehat{T}_{i,j}(\mathbf{z}^{(t)})\left(\widehat{\lambda}_{i,j}(\mathbf{u}^{(v)}) - \widehat{\lambda}_{i,j}(\mathbf{u}^{(0)})\right)\right].
\end{aligned}$$

Here the base measures Q_is are canceled out. Let $\bar{\lambda}(\mathbf{u}) = \lambda(\mathbf{u}) - \lambda(\mathbf{u}^{(0)})$. The above equation can be expressed, with inner products, as follows

$$\langle \mathbf{T}(\mathbf{h}^{(t)}), \bar{\lambda}\rangle + \sum_i \log\frac{Z_i(\mathbf{u}^{(0)})}{Z_i(\mathbf{u}^{(v)})} = \langle \widehat{\mathbf{T}}(\mathbf{z}^{(t)}), \bar{\widehat{\lambda}}\rangle + \sum_i \log\frac{\widehat{Z}_i(\mathbf{u}^{(0)})}{\widehat{Z}_i(\mathbf{u}^{(v)})}, \quad \forall\, v, 1 \leqslant v \leqslant dm.$$

Combine dm equations together and we can rewrite them in matrix equation form as following

$$\mathbf{L}^\top \mathbf{T}(\mathbf{h}^{(t)}) = \widehat{\mathbf{L}}^\top \widehat{\mathbf{T}}(\mathbf{z}^{(t)}) + \mathbf{b}.$$

Here $b_v = \sum_{i=1}^{d} \log\frac{\widehat{Z}_i(\mathbf{u}^{(0)})Z_i(\mathbf{u}^{(v)})}{\widehat{Z}_i(\mathbf{u}^{(v)})Z_i(\mathbf{u}^{(0)})}$. We can multiply $\mathbf{L}^\top$'s inverse with both sized of the equation,

$$\mathbf{T}(\mathbf{h}^{(t)}) = \mathbf{A}\widehat{\mathbf{T}}(\mathbf{z}^{(t)}) + \mathbf{c}. \tag{33}$$

Here $\mathbf{A} = \mathbf{L}^{-1\top}\widehat{\mathbf{L}}^\top$, and $\mathbf{c} = \mathbf{L}^{-1\top}\mathbf{b}$. By Lemma 1 from [15], there exist m distinct values $h_1^{(t),i}$ to $h_m^{(t),i}$ such that $\left[\frac{dT_i}{dh^{(t),i}}(h_1^{(t),i}), ..., \frac{dT_i}{dh^{(t),i}}(h_m^{(t),i})\right]$ are linearly independent in $\mathbb{R}^m$, for all $1 \leqslant i \leqslant d$. Define m vectors $\mathbf{h}_v^{(t)} = [h_v^{(t),1}, ..., h_v^{(t),d}]$ from points given by this lemma. We obtain the following Jacobian matrix

$$\mathbf{Q} = \left[\mathbf{J}_{\mathbf{T}}(\mathbf{h}_1^{(t)}), ..., \mathbf{J}_{\mathbf{T}}(\mathbf{h}_m^{(t)})\right],$$

where each entry is the Jacobian of size $dm \times d$ from the derivative of Eq. (33) regarding the m vectors $\{\mathbf{h}_j^{(t)}\}_{j=1}^m$. Hence $\mathbf{Q}$ is a $dm \times dm$ invertible by the lemma and the fact that each component of $\mathbf{T}$ is univariate. We can construct a corresponding matrix $\widehat{\mathbf{Q}}$ with the Jacobian of $\widehat{\mathbf{T}}(\mathbf{g}^{-1} \circ \mathbf{f}_t^{-1}(\mathbf{h}^{(t)}))$ computed at the same points and get

$$\mathbf{Q} = \mathbf{A}\widehat{\mathbf{Q}}.$$

Here $\widehat{\mathbf{Q}}$ and $\mathbf{A}$ are both full rank as $\mathbf{Q}$ is full rank. □

According to Theorem 1, the proposed model not only can identify global latent factors, but also identify the latent factors for each section with enough auxiliary information. VFG provides a potential approach to learn the latent hierarchical structures from datasets.