Unsupervised Learning with Truncated Gaussian Graphical Models

Qinliang Su, Xuejun Liao, Chunyuan Li, Zhe Gan and Lawrence Carin
Department of Electrical & Computer Engineering
Duke University
Durham, NC 27708-0291

Abstract

Gaussian graphical models (GGMs) are widely used for statistical modeling, because of ease of inference and the ubiquitous use of the normal distribution in practical approximations. However, they are also known for their limited modeling abilities, due to the Gaussian assumption. In this paper, we introduce a novel variant of GGMs, which relaxes the Gaussian restriction and yet admits efficient inference. Specifically, we impose a bipartite structure on the GGM and govern the hidden variables by *truncated* normal distributions. The nonlinearity of the model is revealed by its connection to rectified linear unit (ReLU) neural networks. Meanwhile, thanks to the bipartite structure and appealing properties of truncated normals, we are able to train the models efficiently using contrastive divergence. We consider three output constructs, accounting for real-valued, binary and count data. We further extend the model to deep constructions and show that deep models can be used for unsupervised pre-training of rectifier neural networks. Extensive experimental results are provided to validate the proposed models and demonstrate their superiority over competing models.

Introduction

Gaussian graphical models (GGMs) have been widely used in practical applications (Honorio et al. 2009; Liu and Willsky 2013; Oh and Deasy 2014; Meng, Eriksson, and Hero 2014) to discover statistical relations of random variables from empirical data. The popularity of GGMs is largely attributed to the ubiquitous use of normal-distribution approximations in practice, as well as the ease of inference due to the appealing properties of multivariate normal distributions. On the downside, however, the Gaussian assumption prevents GGMs from being applied to more complex tasks, for which the underlying statistical relations are inherently non-Gaussian and nonlinear. It is true for many models that, by adding hidden variables and integrating them out, a more expressive distribution can be obtained about the visible variables; such models include Boltzmann machines (BMs) (Ackley, Hinton, and Sejnowski 1985), restricted BMs (RBMs) (Hinton 2002; Hinton, Osindero, and Teh 2006; Salakhutdinov and Hinton 2009), and sigmoid belief networks (SBNs) (Neal 1992). Unfortunately, this approach does not work for GGMs since the marginal distribution of visible variables always remains Gaussian no matter how many hidden variables are added (the marginals of a multivariate Gaussian distribution are still Gaussian).

Many efforts have been devoted to enhancing the representational versatility of GGMs. In (Frey 1997; Frey and Hinton 1999), nonlinear Gaussian belief networks were proposed, with explicit nonlinear transformations applied on random variables to obtain nonlinearity. More recently, (Su et al. 2016) proposed to employ truncated Gaussian hidden variables to implicitly introduce nonlinearity. An important advantage of truncation over transformation is that many nice properties of GGMs are preserved, which can be exploited to facilitate inference of the model. However, the models in (Su et al. 2016; Frey 1997; Frey and Hinton 1999) all have a directed graphical structure, for which it is difficult to estimate the posteriors of hidden variables due to the "explaining away" effect inherent in directed graphical models. As a result, mean-field variational Bayesian (VB) analysis was used. It is well known that, apart from the scalability issue, the independence assumption in mean-field VB is often too restrictive to capture the actual statistical relations. Moreover, (Su et al. 2016) is primarily targeted at supervised learning and only considered regression and classification tasks.

We consider an *undirected* GGM with truncated hidden variables. This serves as a counterpart of the directed model in (Su et al. 2016), and it is particularly useful for unsupervised learning. Conditional dependencies are encoded in the graph structure of undirected graphical models. We impose a bipartite structure on the graph, such that it contains two layers (one hidden and one visible) and only has inter-layer connections, leading to a model termed a restricted truncated GGM (RTGGM). In RTGGM, visible variables are conditionally independent given the hidden variables, and vice versa. By exploiting the conditional independencies as well as the appealing properties of truncated normals, we show that the model can be trained efficiently using contrastive divergence (CD) (Hinton 2002). This makes a striking contrast to the directed model in (Su et al. 2016), where the conditionally-independent properties do not exist and inference is done based on mean-field VB approximation.

Although the variables in an RTGGM are conditionally independent, their marginal distributions are flexible enough to model many interesting data. Truncated real observations

Copyright © 2017, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

(e.g., nonnegative) are naturally handled by the RTGGM. We also develop three variants of the basic RTGGM, appropriate for modeling real, binary or count data. It is shown that all variants can also be trained efficiently by the CD algorithm. Furthermore, we extend two-layer RTGGMs to deep models, by stacking multiple RTGGMs together, and show that the deep models can be trained in a layer-wise manner. To evaluate the performance of the proposed models, we have also developed methods to estimate their partition functions, based on annealed importance sampling (AIS) (Salakhutdinov and Murray 2008; Neal 2001). Extensive experimental results are provided to validate the advantages of the RTGGM models.

Related Work

The proposed RTGGM is a new member of the GGM family, and it is also closely related to the RBM (Hinton 2002). One of the main differences between the two models is their inherent nonlinearities. In an RTGGM, the visible and hidden variables are related through smoothed ReLU functions, while they are related by sigmoid functions in an RBM. The ReLU is used extensively in neural networks and has achieved tremendous success due to its training properties (Jarrett et al. 2009). In light of this, there have been many efforts devoted to bringing the ReLU into the RBM formalism. For example, (Nair and Hinton 2010) proposed to replace binary hidden units with a rectified Gaussian approximation. Although an ReLU-like nonlinearity is induced, the proposed model is only specified by two conditional distributions, lacking a joint-distribution description. On the other hand, (Ravanbakhsh et al. 2016) proposed to use Exponential Family Harmoniums (EFH) (Welling, Rosen-Zvi, and Hinton 2004) and Bregman divergence to incorporate different monotonic nonlinearities into the RBM. The model preserves a joint-distribution description, but their conditional distributions are complicated and do not admit exact and efficient sampling. To overcome this, they need to approximate the conditional distributions as Gaussian and then sample from the approximate distributions. In contrast to the above models, the proposed RTGGM not only maintains an explicit joint distribution, but also preserves simple conditional distributions (truncated normals), allowing exact and efficient sampling. Moreover, because of the explicit joint distribution and the easily-sampled conditional distributions, we are able to estimate the partition function of the RTGGM, for performance evaluation. However, it is not clear how to estimate the partition function for models in (Nair and Hinton 2010; Ravanbakhsh et al. 2016). Interestingly, we also note that the smoothed ReLU associated with the proposed RTGGM has similarities with the leaky ReLU (He et al. 2015), as both have small nonzero slopes for negative inputs.

Formulation of Restricted-Truncated Gaussian Graphical Models

Basic Model

Let $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{h} \in \mathbb{R}^m$ denote the visible and hidden variables, respectively. The joint probability distribution of

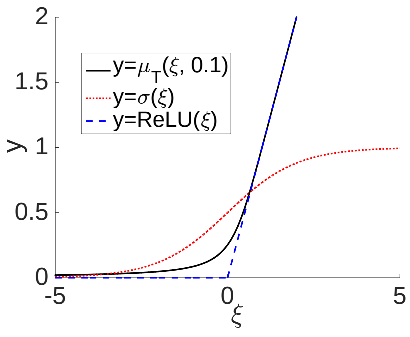

Figure 1: $\mu_T(\xi, 0.1)$ vs $\sigma(\xi) = (1 + e^{-\xi})^{-1}$ and $\mathrm{ReLU}(\xi) = \max(0, \xi)$.

an RTGGM is defined as

$$p(\mathbf{x}, \mathbf{h}; \boldsymbol{\Theta}) = \frac{1}{Z} e^{-E(\mathbf{x},\mathbf{h})} \mathbb{I}(\mathbf{x} \geq 0) \mathbb{I}(\mathbf{h} \geq 0), \qquad (1)$$

where $\mathbb{I}(\cdot)$ is the indicator function, $E(\mathbf{x}, \mathbf{h})$ is an energy function defined as

$$\begin{aligned} E(\mathbf{x}, \mathbf{h}) \triangleq \frac{1}{2} \big(&\mathbf{x}^T \mathrm{diag}(\mathbf{a})\, \mathbf{x} + \mathbf{h}^T \mathrm{diag}(\mathbf{d})\, \mathbf{h} \\ &-2\mathbf{x}^T \mathbf{W} \mathbf{h} - 2\mathbf{b}^T \mathbf{x} - 2\mathbf{c}^T \mathbf{h}\big), \qquad (2) \end{aligned}$$

Z is the partition function, the superscripted T denotes matrix transpose, and $\boldsymbol{\Theta} \triangleq \{\mathbf{W}, \mathbf{a}, \mathbf{d}, \mathbf{b}, \mathbf{c}\}$ collects all model parameters. The joint distribution in (1) can be equivalently written as

$$p(\mathbf{x}, \mathbf{h}; \boldsymbol{\Theta}) = \mathcal{N}_T \left([\mathbf{x}^T, \mathbf{h}^T]^T \,\middle|\, \boldsymbol{\mu}, \mathbf{P}^{-1} \right), \qquad (3)$$

where $\mathcal{N}_T(\cdot)$ represents the *truncated* normal distribution whose nonzero probability density concentrates in the positive orthant, $\mathbf{P} \triangleq \begin{bmatrix} \mathrm{diag}(\mathbf{a}) & -\mathbf{W} \\ -\mathbf{W}^T & \mathrm{diag}(\mathbf{d}) \end{bmatrix} \succ 0$ and $\boldsymbol{\mu} \triangleq \mathbf{P}^{-1} \begin{bmatrix} \mathbf{b} \\ \mathbf{c} \end{bmatrix}$. Because of the diagonal matrices $\mathrm{diag}(\mathbf{a})$ and $\mathrm{diag}(\mathbf{d})$ in (2), we have the conditional distributions as

$$p(\mathbf{x}|\mathbf{h}; \boldsymbol{\Theta}) = \prod_{i=1}^{n} \mathcal{N}_T \left(x_i \,\middle|\, \frac{1}{a_i} [\mathbf{W}\mathbf{h} + \mathbf{b}]_i, \frac{1}{a_i} \right), \qquad (4)$$

$$p(\mathbf{h}|\mathbf{x}; \boldsymbol{\Theta}) = \prod_{j=1}^{m} \mathcal{N}_T \left(h_j \,\middle|\, \frac{1}{d_j} [\mathbf{W}^T\mathbf{x} + \mathbf{c}]_j, \frac{1}{d_j} \right), \qquad (5)$$

where $[\mathbf{z}]_i$ and z_i both represent the i-th element of vector $\mathbf{z}$. Equations (4) and (5) show that the visible variables are conditionally independent given the hidden variables, and vice versa.

By the properties of univariate truncated normal distributions (Johnson, Kotz, and Balakrishnan 1994), the conditional expectation is given by $\mathbb{E}[h_j|\mathbf{x}] = \mu_T(\frac{1}{d_j}[\mathbf{W}^T\mathbf{x} + \mathbf{c}]_j, \frac{1}{d_j})$, where

$$\mu_T(\xi, \lambda^2) \triangleq \xi + \lambda\, \phi(\xi/\lambda) \,/\, \Phi(\xi/\lambda) \qquad (6)$$

is the mean of $\mathcal{N}_T(x|\xi, \lambda^2)$ and it serves as the nonlinearity used in the RTGGM; $\phi(\cdot)$ and $\Phi(\cdot)$ are respectively the probability density function (pdf) and cumulative distribution function (cdf) of the standard normal distribution. Shown in Figure 1 is $\mu_T(\xi, \lambda^2)$ as a function of ξ for $\lambda^2 = 0.1$, along with the sigmoidal and ReLU activation function, for comparison. It is observed that $\mu_T(\cdot, 0.1)$ behaves similar to the ReLU nonlinearity, and deviates significantly from the sigmoidal nonlinearity used in RBMs.

Variants

Truncating the hidden variables in the RTGGM is essential to maintain model expressiveness. In the basic RTGGM above, the visible variables are also truncated to obtain symmetry, but this is not necessary. The visible domain can be changed to match the type of data in an application. Below we present three variants of the basic RTGGM, which deal with real, binary, and count data. In all cases, $p(\mathbf{x}|\mathbf{h};\boldsymbol{\Theta})$ in (4) is modified, but $p(\mathbf{h}|\mathbf{x},\boldsymbol{\Theta})$ remains as in (5), and thus the ReLU nonlinearity is preserved.

Real-Valued Data The joint distribution in this case is $p(\mathbf{x},\mathbf{h};\boldsymbol{\Theta}) = \frac{1}{Z}e^{-E(\mathbf{x},\mathbf{h})}\mathbb{I}(\mathbf{h} \geq \mathbf{0})$. The conditional distribution of the data changes to

$$p(\mathbf{x}|\mathbf{h};\boldsymbol{\Theta}) = \prod_{i=1}^{n} \mathcal{N}\left(x_i \,\middle|\, \frac{1}{a_i}[\mathbf{W}\mathbf{h}+\mathbf{b}]_i, \frac{1}{a_i}\right). \quad (7)$$

Binary Data When each component of $\mathbf{x}$ is in $\{0,1\}$, the quadratic term $\mathbf{x}^T\text{diag}(\mathbf{a})\mathbf{x}$ is dropped from the energy function $E(\mathbf{x},\mathbf{h})$ and the domain restriction is changed from $\mathbb{I}(\mathbf{x} \geq \mathbf{0})$ to $\mathbb{I}(\mathbf{x} \in \{0,1\}^n)$. The conditional in (4) becomes $p(\mathbf{x}|\mathbf{h};\boldsymbol{\Theta}) = \prod_{i=1}^{n} p(x_i|\mathbf{h};\boldsymbol{\Theta})$, with

$$p(x_i = 1|\mathbf{h};\boldsymbol{\Theta}) = \frac{\exp\{[\mathbf{W}\mathbf{h}+\mathbf{b}]_i\}}{1+\exp\{[\mathbf{W}\mathbf{h}+\mathbf{b}]_i\}}. \quad (8)$$

Count Data Without loss of generality, we describe the count data model in the context of topic modeling. Following (Hinton and Salakhutdinov 2009), we employ $N \times 1$ "one-hot" vectors (a one-hot vector is a vector of all 0's except for a single 1) to represent the words in a vocabulary of size N. A document of size K is then represented by a matrix $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_K]$, where each column is a $N \times 1$ one-hot vector. We define an energy function $E(\mathbf{X},\mathbf{h}) \triangleq \frac{1}{2}(\mathbf{h}^T\text{diag}(\mathbf{d})\mathbf{h} - 2\hat{\mathbf{x}}^T\mathbf{W}\mathbf{h} - 2\mathbf{b}^T\hat{\mathbf{x}} - 2K\mathbf{c}^T\mathbf{h})$ with $\hat{\mathbf{x}} \triangleq \sum_{k=1}^{K}\mathbf{x}_k$ understood as a count vector. The energy function above reduces to that of replicated softmax (Hinton and Salakhutdinov 2009) if the quadratic term is dropped and $\mathbf{h}$ is restricted to $\mathbf{h} \in \{0,1\}^m$. The conditional in (4) is accordingly modified to $p(\mathbf{X}|\mathbf{h};\boldsymbol{\Theta}) = \prod_{i=1}^{N}\prod_{k=1}^{K} p([\mathbf{x}_k]_i|\mathbf{h};\boldsymbol{\Theta})$, with

$$p([\mathbf{x}_k]_i = 1|\mathbf{h};\boldsymbol{\Theta}) = \frac{\exp\{[\mathbf{W}\mathbf{h}+\mathbf{b}]_i\}}{\sum_{j=1}^{N}\exp\{[\mathbf{W}\mathbf{h}+\mathbf{b}]_j\}}. \quad (9)$$

Model Training

When training an RTGGM one is concerned with finding the $\boldsymbol{\Theta}$ that maximizes the log-likelihood $\widetilde{\mathcal{L}}(\boldsymbol{\Theta};\mathcal{X}) = \sum_{\mathbf{x}\in\mathcal{X}} \mathcal{L}(\boldsymbol{\Theta};\mathbf{x})$ given the training data set $\mathcal{X}$, where $\mathcal{L}(\boldsymbol{\Theta};\mathbf{x}) = \log\int_0^{+\infty} p(\mathbf{x},\mathbf{h};\boldsymbol{\Theta})d\mathbf{h}$ is the contribution from a single data sample, and $\int_0^{+\infty} d\mathbf{h}$ is a shorthand for the multiple integral with respect to (w.r.t.) the components in $\mathbf{h}$. It is known that $\frac{\partial\mathcal{L}(\boldsymbol{\Theta};\mathbf{x})}{\partial\boldsymbol{\Theta}} = \mathbb{E}[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial\boldsymbol{\Theta}}] - \mathbb{E}[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial\boldsymbol{\Theta}}|\mathbf{x}]$. The first term involves expectation of the energy function w.r.t. $p(\mathbf{x},\mathbf{h})$, which is intractable because of the difficulty of computing the normalizing constant of $p(\mathbf{x},\mathbf{h})$. Fortunately, $p(\mathbf{x}|\mathbf{h})$ and $p(\mathbf{h}|\mathbf{x})$ are both simple distributions, allowing a Gibbs sampler to efficiently sample from $p(\mathbf{x},\mathbf{h})$. Starting with $\mathbf{x}^{(0)} = \mathbf{x}$, the Gibbs sampler generates a chain of samples, $(\mathbf{x}^{(0)},\mathbf{h}^{(1)},\mathbf{x}^{(1)},\ldots,\mathbf{h}^{(k)},\mathbf{x}^{(k)})$, where $\mathbf{h}^{(t)} \sim p(\mathbf{h}|\mathbf{x}^{(t-1)};\boldsymbol{\Theta})$ and $\mathbf{x}^{(t)} \sim p(\mathbf{x}|\mathbf{h}^{(t)};\boldsymbol{\Theta})$. The contrastive divergence uses the first and last sample of $\mathbf{x}$ in the chain, i.e., $\mathbf{x}^{(0)}$ (which is a datum) and $\mathbf{x}^{(k)}$, to from an estimate of the expected gradient,

$$\frac{\partial\mathcal{L}(\boldsymbol{\Theta};\mathbf{x})}{\partial\boldsymbol{\Theta}} \approx \mathbb{E}\left[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial\boldsymbol{\Theta}}\middle|\mathbf{x}^{(k)}\right] - \mathbb{E}\left[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial\boldsymbol{\Theta}}\middle|\mathbf{x}^{(0)}\right]. \quad (10)$$

Note that $p(\mathbf{h}|\mathbf{x};\boldsymbol{\Theta})$ is always a truncated normal distribution as shown in (5), while $p(\mathbf{x}|\mathbf{h};\boldsymbol{\Theta})$ is constituted according to (4), (7), (8), or (9), depending on the type of data $\mathbf{x}$. For the basic RTGGM, we have $\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial w_{ij}} = x_i h_j$, $\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial a_i} = \frac{1}{2}x_i^2$, $\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial b_i} = x_i$, $\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial c_j} = h_j$, and $\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial d_j} = \frac{1}{2}h_j^2$. It can be seen that, to estimate $\frac{\partial\mathcal{L}(\boldsymbol{\Theta};\mathbf{x})}{\partial\boldsymbol{\Theta}}$, one only needs to know the conditional expectations $\mathbb{E}[h_i|\mathbf{x} = \mathbf{x}^{(s)}]$ and $\mathbb{E}[h_i^2|\mathbf{x}^{(s)}]$ for $s = 0, k$. It follows from (6) that $\mathbb{E}\left[h_j|\mathbf{x}^{(s)}\right] = \mu_T(\frac{[\mathbf{W}^T\mathbf{x}^{(s)}+\mathbf{c}]_j}{d_j}, \frac{1}{d_j})$. To compute $\mathbb{E}\left[h_j^2|\mathbf{x}^{(s)}\right]$, we use the formula $\mathbb{E}\left[h_j^2|\mathbf{x}^{(s)}\right] = \mathbb{E}\left[h_j|\mathbf{x}^{(s)}\right]^2 + \text{Var}[h_j|\mathbf{x}^{(s)}]$, where

$$\text{Var}\left[h_j|\mathbf{x}^{(s)}\right] = \frac{1}{d_j}\left(1 - \beta_j\frac{\phi(\beta_j)}{\Phi(\beta_j)} - \frac{\phi^2(\beta_j)}{\Phi^2(\beta_j)}\right) \quad (11)$$

according to (Johnson, Kotz, and Balakrishnan 1994), where $\beta_i \triangleq \frac{[\mathbf{W}^T\mathbf{x}^{(s)}+\mathbf{c}]_j}{\sqrt{d_j}}$. The gradients for the variant RTGGM models can be estimated similarly. With these estimated gradients, the model parameters $\boldsymbol{\Theta}$ can be updated using stochastic optimization algorithms.

One challenge in training RTGGMs is how to efficiently sample from truncated normal distributions. Fortunately, because the variables in an RTGGM are conditionally independent, we only need to sample from univariate truncated normals, and such sampling has been investigated extensively. Many efficient algorithms have been proposed (Chopin 2011; Robert 1995). Another challenge is how to efficiently calculate the ratio $\frac{\phi(\mu)}{\Phi(\mu)}$ in (6) and (11). Direct calculation is expensive due to the integration involved in $\Phi(\mu)$. To compute it cheaply, we adopt the approach in (Su et al. 2016), taking into consideration the approximations based on asymptotic expansions of the Gaussian hazard function.

Partition Function Estimation

To evaluate model performance, we desire the partition function Z. By exploiting the bipartite structure in an RTGGM as well as the appealing properties of truncated normals, we use annealed importance sampling (AIS) (Salakhutdinov and Murray 2008; Neal 2001) to estimate Z. We here only focus on the RTGGM with binary data; details for the other data types are provided in the Supplementary Material. The joint distribution of the RTGGM for binary data can be represented as $p(\mathbf{x}, \mathbf{h}; \boldsymbol{\Theta}) = \frac{1}{Z} e^{-\frac{1}{2}\left(\|\mathbf{D}^{\frac{1}{2}}\mathbf{h}\|^2 - 2\mathbf{x}^T\mathbf{W}\mathbf{h} - 2\mathbf{b}^T\mathbf{x} - 2\mathbf{c}^T\mathbf{h}\right)} I(\mathbf{x} \in \{0,1\}^n) I(\mathbf{h} \geq \mathbf{0})$. After integrating out the hidden variable $\mathbf{h}$, we obtain $p(\mathbf{x}; \boldsymbol{\Theta}) = \frac{1}{Z} p^*(\mathbf{x}; \boldsymbol{\Theta})$, where

$$p^*(\mathbf{x}; \boldsymbol{\Theta}) \triangleq e^{\mathbf{b}^T\mathbf{x}} \times \prod_{j=1}^{m} \frac{1}{\sqrt{d_j}} \frac{\Phi\left(\frac{[\mathbf{W}^T\mathbf{x}+\mathbf{c}]_j}{\sqrt{d_j}}\right)}{\phi\left(\frac{[\mathbf{W}^T\mathbf{x}+\mathbf{c}]_j}{\sqrt{d_j}}\right)}. \quad (12)$$

Since $p^*(\mathbf{x}; \boldsymbol{\Theta})$ is in closed-form, we only need calculate the partition function Z to obtain $p(\mathbf{x}; \boldsymbol{\Theta})$.

Following the AIS procedure (Salakhutdinov and Murray 2008; Neal 2001), we define two distributions $p_A(\mathbf{x}, \mathbf{h}^A) = \frac{1}{Z_A} e^{-E_A(\mathbf{x}, \mathbf{h}^A)}$ and $p_B(\mathbf{x}, \mathbf{h}^B) = \frac{1}{Z_B} e^{-E_B(\mathbf{x}, \mathbf{h}^B)}$, where $E_A(\mathbf{x}, \mathbf{h}^A) \triangleq \frac{1}{2}(\|\text{diag}^{\frac{1}{2}}(\mathbf{d})\mathbf{h}^A\|^2 - 2\mathbf{b}^T\mathbf{x})$ and $E_B(\mathbf{x}, \mathbf{h}^B) \triangleq \frac{1}{2}(\|\text{diag}^{\frac{1}{2}}(\mathbf{d})\mathbf{h}^B\|^2 - 2\mathbf{x}^T\mathbf{W}\mathbf{h}^B - 2\mathbf{b}^T\mathbf{x} - 2\mathbf{c}^T\mathbf{h}^B)$. By construction, $p_0(\mathbf{x}, \mathbf{h}^A, \mathbf{h}^B) = p_A(\mathbf{x}, \mathbf{h}^A)$ and $p_K(\mathbf{x}, \mathbf{h}^A, \mathbf{h}^B) = p_B(\mathbf{x}, \mathbf{h}^B)$. The partition function of $p_A(\mathbf{x}, \mathbf{h}^A)$ is given by $Z_A = \prod_{i=1}^{n}(1 + e^{b_i^A}) \prod_{j=1}^{m} \frac{1}{\sqrt{d_j}} \frac{\Phi(0)}{\phi(0)}$, and the partition function of $p_B(\mathbf{x}, \mathbf{h}^B)$ can be approximated as (Neal 2001)

$$Z_B \approx \frac{\sum_{i=1}^{M} w^{(i)}}{M} Z_A, \quad (13)$$

where $w^{(i)}$ is constructed from a Markov chain that gradually transits from $p_A(\mathbf{x}, \mathbf{h}^A)$ to $p_B(\mathbf{x}, \mathbf{h}^B)$, with the transition realized via a sequence of intermediate distributions,

$$p_k(\mathbf{x}, \mathbf{h}^A, \mathbf{h}^B) = \frac{1}{Z_k} e^{-(1-\beta_k)E_A(\mathbf{x}, \mathbf{h}^A) - \beta_k E_B(\mathbf{x}, \mathbf{h}^B)}, \quad (14)$$

where $0 = \beta_0 < \beta_1 < \ldots < \beta_K = 1$. In particular, the Markov chain $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)}, \ldots, \mathbf{x}_i^{(K)})$ is simulated as $\mathbf{x}_i^{(0)} \sim p_0(\mathbf{x}_i, \mathbf{h}^A, \mathbf{h}^B)$, $(\mathbf{h}^A, \mathbf{h}^B) \sim p_1(\mathbf{h}^A, \mathbf{h}^B | \mathbf{x}_i{}^{(0)})$, $\mathbf{x}_i^{(1)} \sim p_1(\mathbf{x}_i | \mathbf{h}^A, \mathbf{h}^B)$, $\cdots$, $(\mathbf{h}^A, \mathbf{h}^B) \sim p_K(\mathbf{h}^A, \mathbf{h}^B | \mathbf{x}_i{}^{(K-1)})$ and $\mathbf{x}_i^{(K)} \sim p_K(\mathbf{x}_i | \mathbf{h}^A, \mathbf{h}^B)$. From the chain, a coefficient is constructed as $w^{(i)} = \frac{p_1^*(\tilde{\mathbf{x}}_i{}^{(0)})}{p_0^*(\tilde{\mathbf{x}}_i{}^{(0)})} \frac{p_2^*(\tilde{\mathbf{x}}_i{}^{(1)})}{p_1^*(\tilde{\mathbf{x}}_i{}^{(1)})} \cdots \frac{p_K^*(\tilde{\mathbf{x}}_i{}^{(K-1)})}{p_{K-1}^*(\tilde{\mathbf{x}}_i{}^{(K-1)})}$, where

$$p_k^*(\mathbf{x}) = e^{(1-\beta_k)\mathbf{b}^{AT}\mathbf{x}} \prod_{j=1}^{m} \frac{1}{\sqrt{(1-\beta_k)d_j}} \frac{\Phi(0)}{\phi(0)}$$

$$\times e^{\beta_k \mathbf{b}^T\mathbf{x}} \prod_{j=1}^{m} \frac{1}{\sqrt{\beta_k d_j}} \frac{\Phi\left(\frac{\sqrt{\beta_k}[\mathbf{W}^T\mathbf{x}+\mathbf{c}]_j}{\sqrt{d_j}}\right)}{\phi\left(\frac{\sqrt{\beta_k}[\mathbf{W}^T\mathbf{x}+\mathbf{c}]_j}{\sqrt{d_j}}\right)}. \quad (15)$$

Assuming M independent Markov chains simulated in this way, one obtains $\{w^{(i)}\}_{i=1}^{M}$. Note that the Markov chains can be efficiently simulated, as all involved variables are conditionally independent.

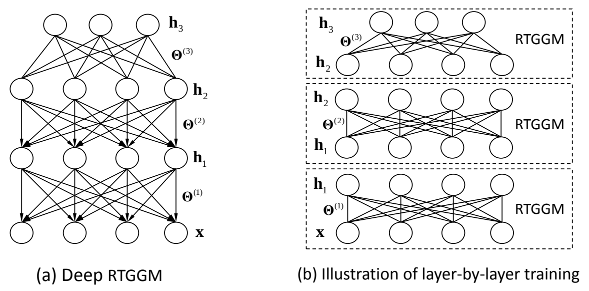

Figure 2: (a) A deep RTGGM with three hidden layers. (b) Layer-wise training of three two-layer RTGGMs.

Extension to Deep Models

The RTGGMs discussed so far consist of a visible layer and a hidden layer. These two-layer models, like RBMs, can be used to construct deep models. A deep RTGGM with L hidden layers, constructed by stacking L two-layer RTGGMs, can be defined by the following joint distribution,

$$p(\mathbf{h}_L, \cdots, \mathbf{h}_1, \mathbf{x}) = p(\mathbf{h}_L, \mathbf{h}_{L-1}) \cdots p(\mathbf{h}_1 | \mathbf{h}_2) p(\mathbf{x} | \mathbf{h}_1), \quad (16)$$

where $p(\mathbf{h}_L, \mathbf{h}_{L-1})$ is the joint distribution of a two-layer RTGGM and $p(\mathbf{h}_{\ell-1} | \mathbf{h}_\ell) = \prod_{i=1}^{M_{\ell-1}} \mathcal{N}_T([\mathbf{h}_{\ell-1}]_i | \frac{1}{a_i^{(\ell)}}[\mathbf{W}^{(\ell)}\mathbf{h}_\ell + \mathbf{b}^{(\ell)}]_i, \frac{1}{a_i^{(\ell)}})$ is the associated conditional distribution. The bottom layer $p(\mathbf{x}|\mathbf{h}_1)$ could be defined by truncated normal, normal, binary or count distributions, depending on the data type. A deep RTGGM with three hidden layers is illustrated in the left panel of Figure 2.

Similar to a DBN constructed from RBMs (Hinton, Osindero, and Teh 2006), a deep RTGGM can be trained in a layer-wise fashion. Specifically, we first train the bottom layer by simply treating it as an RTGGM, using the CD-based ML algorithm described above. We then compute the conditional expectation $\mathbb{E}[\mathbf{h}_1|\mathbf{x}]$ from the already-trained bottom RTGGM and use $\mathbb{E}[\mathbf{h}_1|\mathbf{x}]$ as data to train the second layer from the bottom, again treating it as a RTGGM. The layer-wise training procedure proceeds until the top layer is reached, as illustrated in the right panel of Figure 2. Similar to the proofs in (Hinton, Osindero, and Teh 2006), we can prove that the variational lower bound is guaranteed to increase as more layers are added under the layer-wise training.

Besides serving as a generative model, the deep RTGGM can also be used to pretrain a feedforward neural network so as to improve its performance. It is known that, due to the sigmoidal nonlinearity inherent in RBMs, when we use the unsupervised learning result of a DBN to initialize sigmoidal feed-forward neural networks, remarkable improvements are observed, especially in the case of scarce labeled data (Hinton and Salakhutdinov 2006). Due to the similarity between the nonlinearity in RTGGMs and $ReLU(\cdot)$, we can also use the deep RTGGM learned unsupervisedly to initialize ReLU feedforward neural networks.

Experiments

We report experimental results of the RTGGM models on various publicly available data sets, including binary, count and real-valued data, and compare them to competing models. For all RTGGM models considered below, we use $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(25)}$ to get a CD-based gradient estimate and then use RMSprop (Tieleman and Hinton 2012) to update the model parameters, with the RMSprop delay set to 0.95.

Binary Data

The binarized versions of MNIST and Caltech 101 Silhouettes data sets are considered. MNIST contains 28×28 images of ten handwritten digits, with 60,000 and 10,000 images in the training and testing sets, respectively. Caltech 101 Silhouettes is a set of 28×28 images for the polygon outlines of objects, of which 6364 are used for training and 2307 for testing (Marlin et al. 2010). Two RTGGMs, with 100 and 500 hidden nodes, are trained and tested. The learning rate is set to 10^{-4}. Log-probabilities are estimated based on $100{,}000$ inverse temperatures β_k, uniformly spaced in $[0, 1]$. The final estimate is an average over 100 independent AIS runs.

Tables 1 and 2 summarize the average test log-probabilities on MNIST and Caltech 101 Silhouettes. For comparison, the corresponding results of competing models are also presented. It is seen from Table 1 that the RTGGM with 500 hidden nodes achieves the best performance, significantly outperforming the RBM with the same number of hidden nodes as well as the deep SBN and DBN models. Similar results are observed in Table 2 for the Caltech 101 Silhouettes. The performance gain may be largely attributed to the smooth ReLU nonlinearity brought by truncation, as well as less approximations made in training. The importance of truncation is also revealed in the results of restricted GGMs (RGGM), in which we do not truncate the hidden variables but only impose bipartite structure on GGMs. It can be seen from both Tables 1 and 2 that, without the truncation, RGGMs perform poorly compared to all models. Note that the models in (Nair and Hinton 2010; Ravanbakhsh et al. 2016) are not included here, because their log-probability cannot be computed, as explained earlier.

To demonstrate that the RTGGM is able to capture important statistical relations, we show in Figure 3 samples drawn from the RTGGM trained on MNIST, using a Gibbs sampler with 50000 burn-in samples. We see that the generated digits exhibit large variability and look very similar to real handwritten digits. Moreover, we also use the RTGGM to recover the missing values of an image. It is demonstrated in Figure 3 that, with only the upper-half part of an image presented, the RTGGM can recover the lower-half part reasonably well. The recovery is mostly correct except that "2" is mistaken for "0" in the upper subfigure and "7" is mistaken for "0" in the lower subfigure. However, we notice that the two cases are extremely difficult, in which it would be difficult even for a human to recognize the images based on only the upper-half parts. Finally, we demonstrate in Figure 4 the images drawn from the RTGGM trained on Caltech 101 Silhouettes, using Gibbs sampling with 50,000 burn-in samples. Again, it can be seen that the generated images resemble the training data, showing that the generative model has faithfully captured the features of the training data.

Model	Dim	Test log-prob.
RBM*	500	-86.3
SBN$^\circ$	10-100-200-300-400	-85.4
DBN$^\diamond$	2000-500	-86.2
RGGM	500	-90.2
RTGGM	100	-89.3
RTGGM	500	$\mathbf{-83.2}$

Table 1: Average test log-probability on MNIST. ($\star$) results reported in (Salakhutdinov and Murray 2008) ; ($\circ$) results reported in (Bornschein and Bengio 2015); ($\diamond$) results reported in (Hinton, Osindero, and Teh 2006).

Model	Dim	Test log-prob.
RBM*	500	-114.7
RBM*	4000	-107.7
SBN$^\circ$	10-50-100-300	-113.3
RGGM	500	-350.9
RTGGM	100	-127.8
RTGGM	500	$\mathbf{-105.1}$

Table 2: Average test log-probability on Caltech 101Silhouettes. ($\star$) results reported in (Cho, Raiko, and Ilin 2013); ($\circ$) results reported in (Bornschein and Bengio 2015).

Count Data

Two publicly available corpora are considered: 20News-Groups and Reuters Corpus Volume. The two corpora are preprocessed as in (Hinton and Salakhutdinov 2009). An RTGGM with 50 hidden nodes is trained, using the same setting as in the previous experiment for the learning rate. The perplexity is evaluated over 50 held-out documents, based on the setting used in (Hinton and Salakhutdinov 2009). For each document, we obtain the test log-probability as an average over 100 AIS runs, each using $100{,}000$ inverse temperatures β_k.

Table 3 shows the average test perplexity per word for the RTGGM. For comparison, we also report the perplexities of LDA with 50 and 200 topics, as well as those of the replicated softmax model (RSM) (Hinton and Salakhutdinov 2009) with 50 topics. The RSM is a variant of the RBM that handles count data and it is constructed similarly as the RTGGM for count data. As seen from Table 3, for both corpora, RTGGM-50 performs better than RSM-50 and LDA models. This further demonstrates the performance gains brought about by the smooth ReLU nonlinearity in RTGGMs.

Unsupervised Pre-training of ReLU Neural Networks

As described previously, deep RTGGMs can be used to pre-train multi-layer ReLU neural networks, by exploiting the unlabeled information. In this task, MNIST and NORB datasets are considered, where MNIST is the same as in previous experiments. NORB is a dataset of images from 6 classes

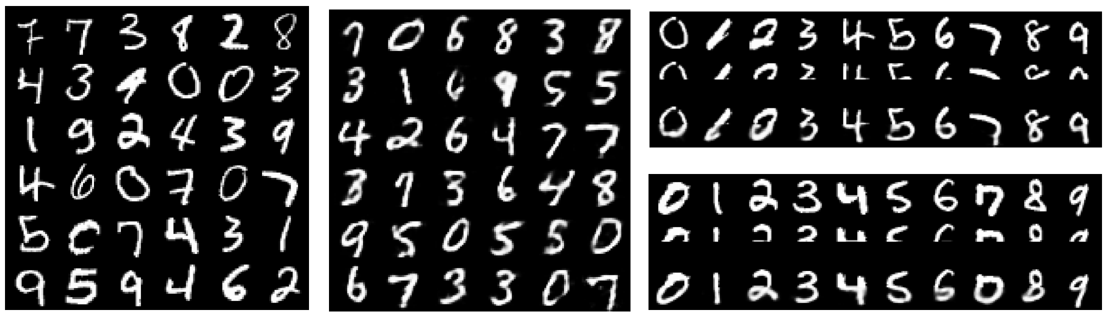

Figure 3: (Left) Samples from the MNIST data set. (Middle) Samples drawn from the RTGGM with 500 hidden nodes. (Right) In each sub-figure, the first row shows the samples from the testing data set, the second row shows the occluded digits presented to the RTGGM, and the bottom row shows the recovered digits. The grayscale values indicate the probabilities.

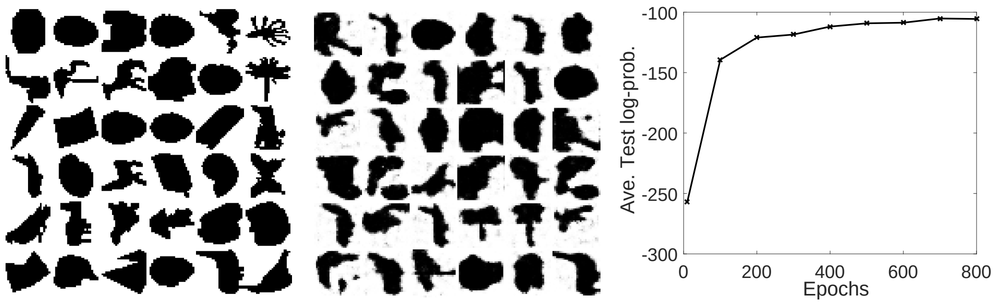

Figure 4: (Left) Samples drawn from Caltech 101 Silhouettes data set. (Middle) Samples drawn from the RTGGM with 500 hidden nodes. (Right) The average test log-probability of the RTGGM with 500 hidden nodes, as a function of learning epoch.

Data set	LDA-50*	LDA-200*	RSM-50*	RTGGM-50
20news	1091	1058	953	**915**
Reuters	1437	1142	988	**934**

Table 3: Average test perplexity per word, on 20newsgroup and Reuters. ($\star$) cited from (Hinton and Salakhutdinov 2009).

Data set	Without pre-train	With pre-train	
		rectified-RBM	RTGGM
MNIST	1.43%	1.33%	**1.17%**
NORB	16.88%	16.43%	**16.12%**

Table 4: Average classification errors achieved by the multi-layer ReLU neural network without pre-training, pre-trained by the method in (Nair and Hinton 2010) (referred to as "rectified-RBM" in the table), and pre-trained by the deep RTGGM.

on cluttered background, partitioned into 291,600 training and 58,230 testing images. We pre-process these images as in (Nair and Hinton 2010). We first train a 1000-1000-1000 deep RTGGM on the unlabeled data, and then use the learned model parameters to initialize a deep ReLU neural network of the same size, which is further trained with the provided labels using RMSprop, with a learning rate of 10^{-5}. The test accuracy is reported in Table 4. For comparison, we also report the results with no pre-training or pre-trained using the rectified-RBM (Nair and Hinton 2010). It is seen from Table 4 that the results of pre-training with the RTGGM performs the best. Note that this is an extension of (Hinton and Salakhutdinov 2006), where the RBM was used to pretrain a sigmoid-based neural network; here we use an RTGGM to pretrain a ReLU-based neural network.

Conclusions

We have introduced a novel variant of the GGM, called restricted truncated GGM (RTGGM), to enhance its representational abilities while preserving its nice (simple inference) properties. The new model is obtained by truncating the variables of an undirected GGM and imposing a bipartite structure on the truncated GGM. It is shown that the truncation brings strong nonlinear representational power to the model, while the bipartite structure enables the model to be trained efficiently using contrastive divergence. Three variants of the RTGGM have been developed to handle real, binary and count data. The two-layer RTGGM has further been extended to produce deep models with multiple hidden layers. Methods have also been developed to estimate the partition function used in evaluating the unsupervised learning performance. Extensive experimental results have demonstrated the superior performance of RTGGMs in unsupervised learning of many types of data, as well as in unsupervised pre-training of feedforward ReLU neural networks.

Acknowledgements

The research reported here was supported by the DOE, NGA, NSF, ONR and by Accenture.

References

[Ackley, Hinton, and Sejnowski 1985] Ackley, D. H.; Hinton, G. E.; and Sejnowski, T. J. 1985. A learning algorithm for boltzmann machines. *Cognitive science* 9(1):147–169.

[Bornschein and Bengio 2015] Bornschein, J., and Bengio, Y. 2015. Reweighted wake-sleep. *ICLR*.

[Cho, Raiko, and Ilin 2013] Cho, K.; Raiko, T.; and Ilin, A. 2013. Enhanced gradient for training restricted boltzmann machines. *Neural computation* 25(3):805–831.

[Chopin 2011] Chopin, N. 2011. Fast simulation of truncated gaussian distributions. *Statistics and Computing* 21(2):275–288.

[Frey and Hinton 1999] Frey, B. J., and Hinton, G. E. 1999. Variational learning in nonlinear gaussian belief networks. *Neural Computation* 11(1):193–213.

[Frey 1997] Frey, B. J. 1997. Continuous sigmoidal belief networks trained using slice sampling. *Advances in Neural Information Processing Systems* 452–458.

[He et al. 2015] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, 1026–1034.

[Hinton and Salakhutdinov 2006] Hinton, G. E., and Salakhutdinov, R. R. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

[Hinton and Salakhutdinov 2009] Hinton, G. E., and Salakhutdinov, R. R. 2009. Replicated softmax: an undirected topic model. In *Advances in neural information processing systems*, 1607–1614.

[Hinton, Osindero, and Teh 2006] Hinton, G. E.; Osindero, S.; and Teh, Y.-W. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18(7):1527–1554.

[Hinton 2002] Hinton, G. E. 2002. Training products of experts by minimizing contrastive divergence. *Neural computation* 14(8):1771–1800.

[Honorio et al. 2009] Honorio, J.; Samaras, D.; Paragios, N.; Goldstein, R.; and Ortiz, L. E. 2009. Sparse and locally constant gaussian graphical models. In *Advances in Neural Information Processing Systems*, 745–753.

[Jarrett et al. 2009] Jarrett, K.; Kavukcuoglu, K.; Ranzato, M.; and LeCun, Y. 2009. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on (ICCV)*, 2146–2153.

[Johnson, Kotz, and Balakrishnan 1994] Johnson, N. L.; Kotz, S.; and Balakrishnan, N. 1994. Continuous univariate distributions, vol. 1-2.

[Liu and Willsky 2013] Liu, Y., and Willsky, A. 2013. Learning gaussian graphical models with observed or latent fvss. In *Advances in Neural Information Processing Systems*, 1833–1841.

[Marlin et al. 2010] Marlin, B. M.; Swersky, K.; Chen, B.; and Freitas, N. D. 2010. Inductive principles for restricted boltzmann machine learning. In *International conference on artificial intelligence and statistics*, 509–516.

[Meng, Eriksson, and Hero 2014] Meng, Z.; Eriksson, B.; and Hero, A. 2014. Learning latent variable gaussian graphical models. In *ICML*, 1269–1277.

[Nair and Hinton 2010] Nair, V., and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 807–814.

[Neal 1992] Neal, R. M. 1992. Connectionist learning of belief networks. *Artificial intelligence* 56(1):71–113.

[Neal 2001] Neal, R. M. 2001. Annealed importance sampling. *Statistics and Computing* 11(2):125–139.

[Oh and Deasy 2014] Oh, J. H., and Deasy, J. O. 2014. Inference of radio-responsive gene regulatory networks using the graphical lasso algorithm. *BMC bioinformatics* 15(Suppl 7):S5.

[Ravanbakhsh et al. 2016] Ravanbakhsh, S.; Póczos, B.; Schneider, J.; Schuurmans, D.; and Greiner, R. 2016. Stochastic neural networks with monotonic activation functions. *AISTATS* 1050:14.

[Robert 1995] Robert, C. P. 1995. Simulation of truncated normal variables. *Statistics and computing* 5(2):121–125.

[Salakhutdinov and Hinton 2009] Salakhutdinov, R., and Hinton, G. E. 2009. Deep boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, 448–455.

[Salakhutdinov and Murray 2008] Salakhutdinov, R., and Murray, I. 2008. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, 872–879. ACM.

[Su et al. 2016] Su, Q.; Liao, X.; Chen, C.; and Carin, L. 2016. Nonlinear statistical learning with truncated gaussian graphical models. In *Proceedings of the 33st International Conference on Machine Learning (ICML-16)*.

[Tieleman and Hinton 2012] Tieleman, T., and Hinton, G. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning* 4.

[Welling, Rosen-Zvi, and Hinton 2004] Welling, M.; Rosen-Zvi, M.; and Hinton, G. E. 2004. Exponential family harmoniums with an application to information retrieval. In *NIPS*, 1481–1488.