1 Proof of Theorem 1 and Theorem 2 in the main paper

In this section, we explain the detail of Theorem 1 and Theorem 2 in the main paper. We start with giving the definition and important properties of unimodal distribution and elliptical distribution. Then we introduce distribution families of our interest, elliptically unimodal distribution. Finally we prove the arguments in Theorem 1 and Theorem 2 in main paper.

1.1 Unimodal Distribution

We first briefly explain the concept of convex unimodal for probability distributions. We refer readers to the book [7] for more detailed discussion on unimodality of distributions, especially in high dimensional space. Some of our definitions directly borrow from [7].

**Definition 1.** A set $S$ is called symmetric about center $c$ if, for all $x$, $c + x \in S \Rightarrow c - x \in S$. A distribution with density $p$ is called symmetric about center $c$ if, for all $x$, $p(c + x) = p(c - x)$.

**Definition 2.** A distribution on $\mathbb{R}^n$ is said to be convex unimodal if it has a density $p$ such that, for every $\eta > 0$, the set $\{x : p(x) > \eta\}$ is convex. Further if for every $\eta > 0$, the set $\{x : p(x) > \eta\}$ is convex and symmetric, then this distribution is called symmetric convex unimodal.

**Lemma 1.** Marginal distributions of symmetric convex unimodal distribution is symmetric convex unimodal.

**Proof.** To prove our lemma, we refer the theorem in book [7] which states that marginal distributions of central convex unimodal distribution is central convex unimodal. Basically, central convex unimodal distribution is symmetric convex unimodal distribution whose symmetric center is origin. Consider a random vector $Y = (Y_1, Y_2)$ has a symmetric convex unimodal distribution with the symmetric center $\mu = (\mu_1, \mu_2)$. Since $Y - \mu$ has a central convex unimodal distribution, $Y_1 - \mu_1$ as the marginal distribution is also central convex unimodal distribution. Thus the distribution of $Y_1$ is symmetric convex unimodal with the symmetric center $\mu_1$.

**Lemma 2.** Condition distributions of convex unimodal distribution is convex unimodal.

**Proof.** Consider a convex unimodal distribution $p(V)$ and its conditional distribution $p(V_S|V_{-S} = \bar{V}_{-S})$ given a subset variables $V_{-S} = \bar{V}_{-S}$. For any $\eta > 0$, consider following sets $C(\eta) = \{V_S : p(V_S|V_{-S} = \bar{V}_{-S}) > \eta\}$, $A = \{V : p(V) > \eta p(V_{-S} = \bar{V}_{-S})\}$ and $B = \{V : V_{-S} = \bar{V}_{-S}\}$. Convex unimodality of $p(V)$ implies that $A$ is convex. Since $B$ is also convex, $A \cap B = \{V = (V_S, V_{-S}) : V_S = \bar{V}_{-S} \text{ and } p(V_S|V_{-S} = \bar{V}_{-S}) = \frac{p(V_S|V_{-S} = \bar{V}_{-S})}{p(V_{-S} = \bar{V}_{-S})} > \eta\}$ is convex. Thus, $C(\eta)$, as the low dimensional projection of $A \cap B$, is convex. Since $C(\eta)$ is convex for any $\eta > 0$, by definition, distribution $p(V_S|V_{-S} = \bar{V}_{-S})$ is convex unimodal.
1.2 Elliptical Distribution

Further, we introduce elliptical distribution. [8] gives a good introduction of elliptical distribution. We refer readers to this paper for more properties of elliptical distribution and the proofs of lemmas we use here.

Definition 3. A random vector $Y$ in space $\mathbb{R}^d$ has an elliptical distribution if its characteristic function has the form,

$$t \mapsto \phi(t) = \exp(it'\mu)\varphi(t'\Sigma t), \quad t \in \mathbb{R}^d.$$  \hspace{1cm} (1)

where $\mu \in \mathbb{R}^d$, $\Sigma \in \mathbb{R}^{d\times d}$ is symmetric and semi-definite and $\varphi : \mathbb{R}^+ \mapsto \mathbb{R}$ is function called characteristic generator. And we denote this distribution as $\mathcal{E}(d, \mu, \Sigma, \phi)$.

Lemma 3 in the following describes the symmetricity of elliptical distributions. This lemma implies that the parameter $\mu$ of the distribution $\mathcal{E}(d, \mu, \Sigma, \phi)$ is actually the mean of the random vector having this distribution. Also distribution $\mathcal{E}(d, \mu, \Sigma, \phi)$ is symmetric about $\mu$.

Lemma 3. A random vector $Y$ has an elliptical distribution $\mathcal{E}(d, \mu, \Sigma, \phi)$ iff $Y \overset{d}{=} \mu + RAU$ \textsuperscript{1} where $R$ is non-negative random variable, $A$ is matrix that satisfies $AA' = \Sigma$, $U$ is a random vector uniformly distributed over $k = rank(\Sigma)$ dimensional unit sphere i.e. $\{u \in \mathbb{R}^k : |u|_2 = 1\}$. And $R$ and $U$ are independent.

In the following lemmas\textsuperscript{2}, we introduce the properties of the marginal and conditional distributions of elliptical distribution. Before that, we introduce more notions in our setting. Let $Y \sim \mathcal{E}(d, \mu, \Sigma, \phi)$ where $\mu = (\mu_1, \mu_2) \in \mathbb{R}^d$, the matrix $\Sigma \in \mathbb{R}^{d\times d}$ is positive semidefinite with $rank(\Sigma) = r$.

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \quad \quad \quad (2)$$

with sub-matrices $\Sigma_{11} \in \mathbb{R}^{k \times k}$, $\Sigma_{12} \in \mathbb{R}^{k \times (d-k)}$, $\Sigma_{21} \in \mathbb{R}^{(d-k) \times k}$, $\Sigma_{22} \in \mathbb{R}^{(d-k) \times (d-k)}$. Further, let $Y = (Y_1, Y_2)$ where $Y_1$ is $k$ dimensional sub-vector of $Y$.

Lemma 4. Distributions of $Y_1$ and $Y_2$ as marginal distributions of $\mathcal{E}(d, \mu, \Sigma, \phi)$ are $\mathcal{E}(d, \mu_1, \Sigma_{11}, \phi_1)$ and $\mathcal{E}(d, \mu_2, \Sigma_{22}, \phi_1)$ respectively which are also elliptical.

Lemma 5. Conditional distribution for $Y_2|Y_1 = y_1$ is elliptical distribution $\mathcal{E}(d, \mu_{2|1}, \Sigma_{2|1}, \phi_{2|1})$, where

$$\mu_{2|1} = \mu_2 + \Sigma_{21}\Sigma_{11}^{-1}(y_1 - \mu_1) \quad \quad \quad (3)$$

$$\Sigma_{2|1} = \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12} \quad \quad \quad (4)$$

1.3 Elliptically Unimodal Distribution

Definition 4. A distribution is said to be elliptically unimodal if it is elliptical, convex unimodal and has a density $p$ with an unique maximum (mode).

Lemma 6. Elliptically unimodal distribution $\mathcal{EU}(d, \mu, \Sigma, \phi)$ is symmetric convex unimodal. Furthermore, the mean vector $\mu$ is the symmetric center and also the unique mode.

Proof. According to Lemma 3, the distribution $\mathcal{EU}(d, \mu, \Sigma, \phi)$ being elliptical implies that it is symmetric about $\mu$, i.e., its density $p$ satisfies $p(\mu + x) = p(\mu - x), \forall x$. For every $\eta > 0$, set $C(\eta) = \{x : p(x) > \eta\}$ is symmetric about $\mu$, since $\mu + x \in C(\eta) \Rightarrow p(\mu + x) > \eta \Rightarrow p(\mu - x) > \eta \Rightarrow \mu - x \in C(\eta)$. Since we already know set $C(\eta)$ is convex due to the fact that the distribution $\mathcal{EU}(d, \mu, \Sigma, \phi)$ is convex unimodal. Thus it is symmetric convex unimodal. We further show that mean vector $\mu$ is the mode. Assume the mode is $\mu + \delta (\delta \neq 0)$, then there is another vector $\mu - \delta$ such that $p(\mu - \delta) = p(\mu + \delta)$, which violates the uniqueness of the mode. Thus any vector other than $\mu$ can not be the mode of the density function. \hfill $\Box$

Lemma 7. Consider $p(V)$ is a elliptically unimodal distribution with the mean vector $\mu$. Its marginal distribution $p(V_S)$ is elliptically unimodal distribution with the mean vector $\mu_S$.

\textsuperscript{1}The distribution equal sign, $\overset{d}{=}$, means the random vectors on both sides have the same distribution.

\textsuperscript{2}The derivations of Lemma 4 and Lemma 5 can be found in section 1.2.4 and 1.2.5 of [8].
**Proof.** By Lemma 4, we know marginal distribution $p(V_S)$ is elliptical and have the mean vector $\mu_S$. By Lemma 1, we know the marginal distribution $p(V_S)$ is symmetric convex unimodal. Combining together, we have that the marginal distribution $p(V_S)$ is elliptically unimodal distribution with mean vector $\mu_S$.

**Lemma 8.** Consider $p(V)$, a elliptically unimodal distribution with mean vector $\mu$. Its conditional distribution $p(V|V_S = \tilde{V}_S)$ is elliptically unimodal distribution. Furthermore, if $\tilde{V}_S = \mu_S$, the mean vector of conditional distribution $p(V_S|V_S = \mu_S)$ is $\mu_S$.

**Proof.** By Lemma 5, we know the conditional distribution $p(V_S|V_S = \tilde{V}_S)$ is elliptical. By Lemma 2, we know the conditional distribution $p(V_S|V_S = \tilde{V}_S)$ is convex unimodal. Thus the conditional distribution $p(V_S|V_S = \tilde{V}_S)$ is also elliptically unimodal. Furthermore, if $\tilde{V}_S = \mu_S$, according to the given Eqn. 3 in Lemma 5, we can derive that the mean vector for $p(V_S|V_S = \tilde{V}_S)$ is $\mu_S$.

### 1.4 Proof of the Theorem

In the following, we restate the arguments of Theorem 1 and Theorem 2 in main paper give the proofs separately.

**Theorem 1.** In the forward inference case, we predict a set of variables $V$ based on the distribution $p(V|X)$ conditioned on given variables $X$. Our prediction $\tilde{V}$ satisfies following greedy property.

\[
\tilde{v}_n = \arg\max_{v_n} p(v_n|X, \tilde{V}), \quad n = 1, \ldots, N
\]

**The arguement of this theorem is:** if the distribution $p(V|X)$ is elliptically unimodal, our prediction is actually global optimum, i.e., $\tilde{V} = \arg\max_V p(V|X)$

**Proof.** Since $p(V|X)$ is elliptically unimodal, say its mode is $V^*$. By Lemma 7, we know for every $n$, the marginal distribution $p(v_n|X)$ is also elliptically unimodal with mode $v_n^*$. Further by Lemma 8, $p(v_n|X)$’s conditional distribution $p(v_n|X, V_{n-1} = V_{n-1}^*)$ is elliptically unimodal with mode $v_n^*$. Thus at first step in forward inference, our prediction for variable $v_1$, $\tilde{v}_1$, satisfies $\tilde{v}_1 = \arg\max_{v_1} p(v_1|X) = v_1^*$. For the following steps, the prediction $\tilde{v}_n$ satisfies $\tilde{v}_n = \arg\max_{v_n} p(v_n|X, V_{n-1} = \tilde{V}_{n-1} = V_{n-1}^*) = v_n^*$. Thus in total, our prediction $\tilde{V} = V^*$.

**Theorem 2.** In hybrid inference case, we predict a subset of variables $V_S$ based on the distribution $p(V_S|X, V_S = \tilde{V}_S)$ with given extra variable $X$. We divide the target variables $V_S$ into two parts: $V_B = V_S \cap \{v_i: i < L\}$ and $V_S = V_S \cap \{v_i: i > L\}$, where $L$ is largest index of variable in $V_S$, i.e., $L = \max_{i \in S} i$. We first do backward inference and assume the result $\tilde{V}_B$ satisfies $\tilde{V}_B = \arg\max_{v_B} p(V_B|X, V_S = \tilde{V}_S)$. Then we use forward prediction to get the prediction $\tilde{V}_F$. The arguement of this theorem is: if the distribution $p(V_S|X, V_S = \tilde{V}_S)$ is elliptically unimodal, our prediction is actually global optimum, i.e., $\tilde{V}_S = (\tilde{V}_B, \tilde{V}_F) = \arg\max_{V_S} p(V_S|X, V_S = \tilde{V}_S)$

**Proof.** We denote the mode of elliptically unimodal distribution $p(V_S|X, V_S = \tilde{V}_S)$ as $V_S^*$. By Lemma 7, the marginal distribution $p(V_B|X, V_S = \tilde{V}_S)$ is an elliptically unimodal distribution with mode $V_B^*$. By Lemma 8, the conditional distribution $p(V_F|X, V_S = \tilde{V}_S, V_B = V_B^*)$ is an elliptically unimodal distribution with mode $V_F^*$. By our assumption, the prediction $\tilde{V}_B = \arg\max_{v_B} p(V_B|X, V_S = \tilde{V}_S) = V_B^*$. Further, since $p(V_F|X, V_S = \tilde{V}_S, V_B = V_B^*)$ is elliptically unimodal, by Theorem 1 we can conclude that the forward prediction actually gives global optimum i.e. $\tilde{V}_F = \arg\max_{v_F} p(V_F|X, V_S = \tilde{V}_S, V_B = V_B^*) = V_F^*$. Thus in total our prediction result $\tilde{V}_S = (\tilde{V}_B, \tilde{V}_F)$ equals to the global optimum $V_S^* = (V_B^*, V_F^*)$.

### 2 Marginal Likelihood of $V_S$

In the paper, we propose to approximate the marginal negative log-likelihood $\mathcal{L}(V_S|X; \theta)$ efficiently and effectively by leveraging the properties of NPN. The process is as follows:

\[
\mathcal{L}(V_S|X; \theta) \approx \sum_{v_n \in S_j} - \log p(v_n|X),
\]

where $- \log p(v_n|X)$ can be computed recursively as follows:
which takes distributions as input. The input distributions will go through layers of linear and nonlinear transformation where 

$$\mu(\cdot)$$

of the variance of the joint distribution of 

different mean and variance, chaining the context information is a vector 

$$x$$

hidden neurons and parameters), the mean and variance of the output 

$$a$$

parameters 

$$\theta$$

corresponding Gaussian NPN would assume 

$$f$$

to produce output distributions. In NPN, all hidden neurons and weights are also distributions expressed in closed form.

Different from vanilla neural networks which usually take deterministic input, NPN is a probabilistic neural network 

2.1 More Background on NPN

In this section, we justify this approximation by showing that if each NPN subnetwork has a single layer, the process 

above computes the mean and variance 

$$V_S$$

exactly (note that since NPN assumes diagonal covariance matrices for the 

output, our process can only compute the diagonal entries of the covariance matrix for 

$$V_S$$

exactly and ignores the off-diagonal entries).

2.2 Proof on the Process of Computing 

In a linear NPN layer, if the input 

$$a$$

is drawn from a distribution 

$$p(a|a_m, a_s)$$

(not necessarily Gaussian) with the mean 

$$a_m$$

and the variance 

$$a_s$$

and the weights 

$$W$$

we ignore biases in this section for simplicity) are drawn from a distribution 

$$p(W|W_m, W_s)$$

with the mean 

$$W_m$$

and the variance 

$$W_s$$

(NPN assumes diagonal covariance matrix for hidden neurons and parameters), the mean and variance of the output 

$$o$$

denoted as 

$$o_m$$

and 

$$o_s$$

, can be computed as:

$$o_m = \mu_\theta(a_m, a_s) = a_m W_m,$$

$$o_s = \sigma_\theta(a_m, a_s) = a_s W_s + a_s(W_m \circ W_m) + (a_m \circ a_m)W_s,$$

where 

$$o_m$$

and 

$$o_s$$

are the mean and variance (diagonal entries of the covariance matrix) of the following distribution:

$$p(o|a_m, a_s, W_m, W_s) = \int p(a|a_m, a_s)p(W|W_m, W_s)p(o|a, W)dadW,$$

where 

$$p(o|a, W)$$

is a Dirac delta distribution centered at 

$$aW$$.

These properties turn out to be the key to efficient computation of marginal negative log-likelihood 

$$L(V_{-S}|X; \theta)$$. 

2.2 Proof on the Process of Computing 

$$L(V_{-S}|X; \theta)$$

Notation: In the following, we denote a vector of scalar variable 

$$(v_1, v_2, \ldots, v_k)$$

as 

$$v_k$$

and assume the high-dimensional context information is a vector 

$$x \in \mathbb{R}^C$$. We will prove that if each NPN subnetwork has one layer and output the correct mean and variance, chaining the 

$$N$$

networks using the process mentioned above will produce the correct mean and variance of the joint distribution of 

$$v_N$$

(given 

$$x$$

). Since 

$$V_{-S} \subseteq V$$

, the process can also give the correct mean and variance for the vector 

$$(v_n)$$

where 

$$v_n \in V_{-S}$$. We assume that the 

$$n$$

-th NPN subnetwork use the mean and variance of 

$$(x, v_{n-1})$$

as input and output the mean and variance of 

$$v_n$$. Specifically, 

$$p(x|\phi_x)$$

is the distribution over 

$$x$$

, and

- \( - \log p(v_1|X) = \frac{\| \mu_1(X) - v_1 \|^2}{2s_{v_1}(X)} + \frac{1}{2} \log s_{v_1}(X). \)
- For \( k > 1 \),
  \[
  - \log p(v_k|X) = \frac{\| \mu_k(X, \widehat{U}_{k-1}) - v_k \|^2}{2s_{v_k}(X, \widehat{U}_{k-1})} + \frac{1}{2} \log s_{v_k}(X, \widehat{U}_{k-1}),
  \]
  where \( \widehat{U}_k = \{ \widehat{u}_1, \ldots, \widehat{u}_k \} \) with \( \widehat{u}_k \) as the estimated mean and variance (output by NPN) of \( v_k \) given \( X \) (note that NPN can take mean-variance pairs as input):
  \[
  \widehat{u}_k = (\mu_{\theta_k}(X, \widehat{U}_{k-1}), s_{\theta_k}(X, \widehat{U}_{k-1}))
  \]

In this section, we justify this approximation by showing that if each NPN subnetwork has a single layer, the process 

above computes the mean and variance 

$$V_S$$

exactly (note that since NPN assumes diagonal covariance matrices for the 

output, our process can only compute the diagonal entries of the covariance matrix for 

$$V_S$$

exactly and ignores the off-diagonal entries).
\(p(w_n|\theta_n)\) is the distribution over the weights \(w_n\) of the \(n\)-th NPN subnetwork. Here \(\phi_x\) and \(\theta_n\) are the parameters for corresponding distributions. We further define the shorthand \(W_k = \{w_i\}_{i=1}^k\) and \(\Theta_k = \{\theta_i\}_{i=1}^k\) for convenience. Vectors such as \(v_0\) are \(\phi_0\) empty vectors, which can be ignored during derivation. \((\cdot, \cdot)\) is used to denote concatenation of vectors. To prevent clutter, we omit all biases \(b\) in the network parameters (note that the theorem still holds with the biases).

If each NPN subnetwork has only one linear layer, Eqn. 6 and Eqn. 7 for the \(n\)-th network can be written as (omitting the bias terms):

\[
\mu_{\theta}(x_m, v_{n-1,m}) = (x_m, v_{n-1,m})w_m^T, \\
s_{\theta}(x_m, v_{n-1,m}) = (x_s, v_{n-1,s})w^T + (x_s, v_{n-1,s})w_m^T + ((x_m, v_{n-1,m}) \circ (x_m, v_{n-1,m}))w_m^T,
\]

(9) (10)

where \((x_m, x_s)\) and \((w_m, w_s)\) are the mean-variance pairs of the distributions \(p(x|\phi_x)\) and \(p(w_n|\theta_n)\), respectively. \((v_{n-1,m}, v_{n-1,s})\) is the mean-variance pair for \(v_{n-1}\).

**Theorem 3.** Assume all \(n\) single-layer NPN subnetworks are correct, namely, the output \((\mu_{\theta_n}(\phi_x, \psi_{n-1}), s_{\theta_n}(\phi_x, \psi_{n-1}))\) of the \(n\)-th subnetwork is the mean and variance of the following distribution (where \(\phi_x\) can be the mean and variance of \(x\) and \(\psi_{n-1}\) can be the mean and variance of \(v_{n-1}\)):

\[
p(v_n|\phi_x, \psi_{n-1}, \theta_n) = \int p(x|\phi_x)p(v_{n-1}|\psi_{n-1})p(w_n|\theta_n)p(v_n|x, v_{n-1}, W_n)dx dv_{n-1} dw_n, 1 \leq n \leq N
\]

(11)

where \(p(v_n|x, v_{n-1}, W_n)\) a Dirac delta distribution centered at \(v_n = (x, v_{n-1})w_n^T\) (computed recursively). Consider the following recursive process (also mentioned at the start of Sec. 2):

- Define the concatenation of tuples \(\hat{u}_k = (\hat{u}_1, \ldots, \hat{u}_k)\) with each tuple \(\hat{u}_k\) as the estimated mean and variance (output by NPN) of \(u_k\) given \(x\) (note that NPN can take mean-variance pairs as input).
- Let \(\hat{u}_1 = (\mu_{\theta_1}(x), s_{\theta_1}(x))^T\).
- For \(1 < k \leq N\), let \(\hat{u}_k = (\mu_{\theta_k}(x, \hat{u}_{k-1}), s_{\theta_k}(x, \hat{u}_{k-1}))^T\).

Then the computed \(\hat{u}_N\) contains the mean and variance (diagonal entries of the covariance matrix) of the joint distribution of all variables \(v_N\):

\[
p(v_N|\phi_x, \Theta_N) = \int p(x|\phi_x)p(W_N|\Theta_N)p(v_N|x, W_N)dx dW_N,
\]

(12)

where \(p(v_N|x, W_N)\) a Dirac delta distribution centered at \(v_N = (v_n)_{n=1}^N\) and \(v_n = (x, v_{n-1})w_n^T\) (computed recursively). Specifically \(\hat{u}_{N,1} = (\hat{u}_{n,1})_{n=1}^N\) (concatenate the first entries of all tuples to a vector) is the mean, and similarly \(\hat{u}_{N,2} = (\hat{u}_{n,2})_{n=1}^N\) is the variance.

**Proof.** We first focus on the mean of \(p(v_N|\phi_x, \Theta_N)\) and prove it by induction. For the base case, according to Eqn. 11, \(\hat{u}_1 = (\mu_{\theta_1}(x), s_{\theta_1}(x))\) is the mean and variance of the distribution:

\[
p(v_1|\phi_x, \Theta_1) = \int p(x|\phi_x)p(W_1|\Theta_1)p(v_1|x, W_1)dx dW_1
\]

(13)

\[
= \int p(x|\phi_x)p(w_1|\theta_1)p(v_1|x, w_1)dx dw_1.
\]

(14)

Assume \(\hat{u}_{n-1,1}\) is the mean of the distribution

\[
p(v_{n-1}|\phi_x, \Theta_{n-1}) = \int p(x|\phi_x)p(W_{n-1}|\Theta_{n-1})p(v_{n-1}|x, W_{n-1})dx dW_{n-1}.
\]

(15)

Then the mean of the distribution

\[
p(v_n|\phi_x, \Theta_n) = \int p(x|\phi_x)p(W_n|\Theta_n)p(v_n|x, W_n)dx dW_n
\]

(16)
can be written as
\[
\int (v_{n-1}, \ x, v_{n-1}) w_n^T \ p(x|\phi) \ p(W_n|\Theta_n) \ p(v_n|x, W_n) \ dx \ dW_n \tag{17}
\]
\[
= \int (v_{n-1}, \ x, v_{n-1}) w_n^T \ p(x|\phi) \ p(W_n|\theta_n) \ p(v_{n-1}|x, W_{n-1}) \ dx \ dW_n
\]
\[
= \int (v_{n-1}, \ x, v_{n-1}) w_n^T \ p(x|\phi) \ p(w_n|\theta_n)(\int p(x|\phi) \ dx) \ p(W_{n-1}|\Theta_{n-1}) \ p(v_{n-1}|x, W_{n-1}) \ dx \ dW_{n-1} dW_n \ dv_{n-1}
\]
\[
= \int (v_{n-1}, \ x, v_{n-1}) w_n^T \ p(x|\phi) \ p(w_n|\theta_n)(\int p(x|\phi) \ p(W_{n-1}|\Theta_{n-1}) \ p(v_{n-1}|x, W_{n-1}) \ dx \ dW_{n-1}) \ dx \ dW_n \ dv_{n-1}
\]
\[
= \int (v_{n-1}, \ x, v_{n-1}) w_n^T \ p(x|\phi) \ p(w_n|\theta_n) \ p(v_{n-1}|\phi, \Theta_{n-1}) \ dx \ dW_n \ dv_{n-1}
\]

According to the case of \((n-1)\) in Eqn. 15, we have the mean of \(p(v_{n-1}|\phi, \Theta_{n-1})\) as
\[
\int v_{n-1} \ p(x|\phi) \ p(w_n|\theta_n) \ p(v_{n-1}|\phi, \Theta_{n-1}) \ dx \ dW_n \ dv_{n-1}
\]
\[
= \left( \int v_{n-1} \ p(v_{n-1}|\phi, \Theta_{n-1}) \ dx \ dW_n \ dv_{n-1} \right) \ p(x|\phi)
\]
\[
= \int \hat{u}_{n-1,1*} \ p(x|\phi) \ p(w_n|\theta_n) \ dx \ dW_n
\]
\[
= \hat{u}_{n-1,1*} \ p(x|\phi) \ p(w_n|\theta_n) \ dx \ dW_n
\]

Besides, we have
\[
\int (x, v_{n-1}) w_n^T \ p(x|\phi) \ p(w_n|\theta_n) \ p(v_{n-1}|\phi, \Theta_{n-1}) \ dx \ dW_n \ dv_{n-1}
\]
\[
= (x_m, v_{n-1, m}) w_n^T
\]
\[
= \mu (x_m, v_{n-1, m})
\]
\[
= \hat{u}_{n,1, m}
\]

Hence the mean of the distribution \(p(v_n|\phi, \Theta_n)\) is:
\[
\int (v_{n-1}, \ x, v_{n-1}) w_n^T \ p(x|\phi) \ p(W_n|\Theta_n) \ p(v_n|x, W_n) \ dx \ dW_n = \hat{u}_{n,1*}
\]
meaning that \(\hat{u}_{n,1*}\) is the mean of \(p(v_N|\phi, \Theta_N)\).

Next we use similar techniques to prove that \(\hat{u}_{N,2*}\) is the variance of \(p(v_N|\phi, \Theta_N)\). For the base case, according to Eqn. 11, \(\hat{u}_1 = (\mu_1(x), s\theta_1(x))\) is the mean and variance of the distribution:
\[
p(v_1|\phi, \Theta_1) = \int p(x|\phi) \ p(W_1|\Theta_1) \ p(v_1|x, W_1) \ dx \ dW_1 \tag{18}
\]
\[
= \int p(x|\phi) \ p(W_1|\theta_1) \ p(v_1|x, W_1) \ dx \ dW_1. \tag{19}
\]

Assume \(\hat{u}_{n-1,1*}\) and \(\hat{u}_{n-1,2*}\) are the mean and variance of the distribution
\[
p(v_{n-1}|\phi, \Theta_{n-1}) = \int p(x|\phi) \ p(W_{n-1}|\Theta_{n-1}) \ p(v_{n-1}|x, W_{n-1}) \ dx \ dW_{n-1}. \tag{20}
\]

Then the covariance matrix of the distribution
\[
p(v_{n}|\phi, \Theta_n) = \int p(x|\phi) \ p(W_n|\Theta_n) \ p(v_n|x, W_n) \ dx \ dW_n \tag{21}
\]
can be written as

\[
\int (v_{n-1}, (x, v_{n-1})w_n^T) - \tilde{u}_{N,1*}^T ((v_{n-1}, (x, v_{n-1})w_n^T) - \tilde{u}_{N,1*}) p(x|\phi_x)p(W_n|\Theta_n)p(v_n|x, W_n)dxdw_n,
\]

\[
= \int (v_{n-1}, (x, v_{n-1})w_n^T) (v_{n-1}, (x, v_{n-1})w_n^T) p(x|\phi_x)p(W_n|\Theta_n)p(v_n|x, W_n)dxdw_n - \tilde{u}_{n,1*}^T \tilde{u}_{n,1*}
\]

\[
= E[diag(v_{n-1}^T (v_{n-1}, (x, v_{n-1})w_n^T)^2)] - E^2[diag(v_{n-1}^T (x, v_{n-1})w_n^T)] + A_n
\]

\[
= diag((\tilde{u}_{n-2,3*}, \tilde{u}_{n,2})) + A_n
\]

\[
= diag(\tilde{u}_{n,2*}) + A_n,
\]

where \((v_{n-1}, (x, v_{n-1})w_n^T)\) is an \(n\)-by-\(n\) matrix, and \(\tilde{u}_{n,1*}\) is the mean of \(p(v_n|\phi_x, \Theta_n)\).

\(v_{n-1}^2\) is the element-wise square of \(v_{n-1}\). \(diag(a)\) denotes the diagonal matrix with values in the vector \(a\) as the diagonal entries. Eqn. 25 is due to Eqn. 20 and the fact the assumption that \(\tilde{u}_{n,2}\) is the variance of the distribution in Eqn. 11. In Eqn. 24, the expectation is over the distribution \(p(W_n|\Theta_n)p(v_n|x, W_n)\), and entries of \(A_n \in \mathbb{R}^{n \times n}\) can be computed recursively as:

\[
A_{n,ij} = \begin{cases} 
0 & 1 \leq i = j \leq n \\
A_{n-1,ij} & i < n \text{ and } j < n \text{ and } i \neq j \\
\hat{u}_{n,2w_{n,m}^{(C+i)}} & i < n \text{ and } j = n \\
\hat{u}_{j,2w_{n,m}^{(C+j)}} & i = n \text{ and } j < n,
\end{cases}
\]

where \(w_{n,m}^{(k)}\) is the \(k\)-th entry of \(w_{n,m}\). Hence the diagonal entries of the covariance matrix for the distribution in Eqn. 21 is \(\tilde{u}_{n,2}\), which completes the proof for the variance part.

**Remark:** Note that the theorem above is general since we do not assume Gaussian distributions. And since \(\hat{u}_{n,2}\) and \(\hat{u}_{N,2*}\) are the mean and variance for the joint distribution Eqn. 12 (corresponding to \(V\)), respectively, the process in Theorem 3 also computes the correct mean and variance for the marginal distribution for \(V_S\). Theorem 3 can also be extended to the case where there is nonlinearity after the linear layer, as long as the mean and variance of the nonlinear layer can be computed exactly, as in most cases of NPN [20].

### 2.3 Marginal MAP

Note that using similar techniques above, BIN can be extended to get all marginal MAPs. For example, to get the marginal MAP of \(p(v_2|X, v_3)\) during inference, we need to marginalize out \(v_1\) and compute \(p(v_2|v_3|X) = p(v_2|X)p(v_3|X, v_2)\). To do this, instead of using a deterministic \(v_1\), we use the output mean and variance \((\mu_{v_1}(X), \sigma_{v_1}(X))\) of the first subnetwork \(p(v_1|X)\) as the input to subnetwork \(p(v_2|X, v_1)\) to get the marginal \(p(v_2|X)\). Marginal \(p(v_3|X, v_2)\) can be obtained from subnetwork \(p(v_3|X, v_1, v_2)\) similarly. Maximizing \(p(v_2|X)p(v_3|X, v_2)\) with a fixed \(v_3\) produces the MAP for \(v_2\).

### 3 More Experimental Results

#### 3.1 Toy Inference Tasks

Besides the toy inference task in the main paper, we examine a slightly more complex toy dataset where \(X\) is also considered (here \(X\) is a scalar for simplicity). We generate 8 data points \(\{(X^{(i)}, v_1^{(i)}, v_2^{(i)})\}_{i=1}^8\) according to \(v_1 = 3X + 1 + \epsilon_1\) and \(v_2 = 0.5v_1 - X + 1 + \epsilon_2\), where \(\epsilon_1\) and \(\epsilon_2\) are sampled from \(\mathcal{N}(0, 1)\). \(X\) is sampled from a uniform distribution \(U(-1, 1)\). We use similar hyperparameters as the first toy task (see the Supplement for details).

Again, we train BIN according to Eqn. 4 in the paper and CBIN according to Eqn. 8 \((J = 1\) and \(V_{S_1} = \{v_1\}\) in the paper. The inference task is to infer \(V_S = \{v_1\}\) given \(X\) and \(V_{S_1} = \{v_2\}\).

Fig. 1(a) and Fig. 1(b) show the contours of \(\mu_{v_2}(X, v_1)\) (predicted mean of \(v_2\)) learned by BIN and CBIN, respectively, with the original training data points. As we can see in Fig. 1, with a given \(X\) there are usually more than 1 local minima of \(\mu_{v_2}(X, v_1)\) with respect to \(v_1\) (there are even 5 when \(X\) is around \(-0.4\)) for BIN while there are much fewer for CBIN. Correspondingly, Fig. 1(c) and Fig. 1(d) show the loss surface of \(L\) with respect to \(V_S = \{v_1\}\) when inferring \(v_1\) given \((X^{(1)}, v_2^{(1)})\). As expected, BIN is easier to get trapped in poor local optima (shown as green
Figure 1: (a) and (b): $\mu_{\theta_2}(X, v_1)$ learned by BIN and CBIN. (c) and (d): corresponding loss surface of $\mathcal{L}$ with respect to $\{v_1\}$ when inferring $v_1$ given $X^{(1)}$ and $v_2^{(1)}$.

Table 1: Accuracy (%) for predicting $V_S$ given $X$ and $V_{-S} = \{v_n\}_{n=1}^3 \setminus V_S$ in the SHHS2 dataset.

<table>
<thead>
<tr>
<th></th>
<th>${v_1}$</th>
<th>${v_2}$</th>
<th>${v_1, v_2}$</th>
<th>${v_1, v_2, v_3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPEN</td>
<td>68.66</td>
<td>67.38</td>
<td>68.33</td>
<td>64.24</td>
</tr>
<tr>
<td>eSPEN</td>
<td>69.43</td>
<td>68.31</td>
<td>68.87</td>
<td>64.85</td>
</tr>
<tr>
<td>SVAE</td>
<td>67.75</td>
<td>66.77</td>
<td>66.95</td>
<td>62.86</td>
</tr>
<tr>
<td>PO</td>
<td>69.74</td>
<td>76.85</td>
<td>64.80</td>
<td>68.13</td>
</tr>
<tr>
<td>RI</td>
<td>70.36</td>
<td>70.16</td>
<td>64.42</td>
<td>65.78</td>
</tr>
<tr>
<td>BIN</td>
<td>78.19</td>
<td>77.54</td>
<td>71.87</td>
<td>72.53</td>
</tr>
<tr>
<td>CBIN</td>
<td><strong>78.50</strong></td>
<td><strong>78.77</strong></td>
<td><strong>72.27</strong></td>
<td><strong>73.91</strong></td>
</tr>
<tr>
<td>Retrain</td>
<td><strong>78.86</strong></td>
<td><strong>78.31</strong></td>
<td><strong>71.45</strong></td>
<td><strong>73.72</strong></td>
</tr>
</tbody>
</table>

Table 2: Accuracy (%) when $V_S = V$ for the SHHS2 dataset.

<table>
<thead>
<tr>
<th></th>
<th>${v_1}$</th>
<th>${v_1, v_2}$</th>
<th>${v_1, v_2, v_3}$</th>
<th>${v_n}_{n=1}^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPEN</td>
<td>-</td>
<td>67.26</td>
<td>65.28</td>
<td>65.35</td>
</tr>
<tr>
<td>eSPEN</td>
<td>-</td>
<td>68.56</td>
<td>65.54</td>
<td>66.13</td>
</tr>
<tr>
<td>SVAE</td>
<td>-</td>
<td>68.14</td>
<td>65.71</td>
<td>65.90</td>
</tr>
<tr>
<td>BIN</td>
<td>-</td>
<td>68.64</td>
<td>66.05</td>
<td><strong>66.32</strong></td>
</tr>
<tr>
<td>CBIN</td>
<td>-</td>
<td><strong>69.04</strong></td>
<td><strong>66.35</strong></td>
<td>66.26</td>
</tr>
<tr>
<td>Retrain</td>
<td><strong>71.21</strong></td>
<td>68.11</td>
<td>65.63</td>
<td>66.09</td>
</tr>
</tbody>
</table>

diamonds) than CBIN. Note that since $\mathcal{L}$ in Fig. 1(c) is a quadratic function of $\mu_{\theta_2}(X, v_1)$ in Fig. 1(a), local minima in Fig. 1(a) do not correspond to local optima in Fig. 1(c).

3.2 Experiments on the SHHS2 Dataset

Table 2 shows the accuracy in forward inference cases where $V_S = V$ with different $V$. We can see that SVAE, SPEN, and eSPEN achieve similar or slightly better accuracy than the retrained specific models which essentially assumes conditional independence between variables in $V$ given $X$. Compared to retrained models, BIN and CBIN consider also the conditional dependence among variables, making the predictions more accurate.

Fig. 2(left) shows the number of inference iterations needed to predict $V_S = \{v_1, v_2\}$ given $V_{-S} = \{v_3\}$ versus number of inner loop iterations during training ($T_{in}$ in Algorithm 1 of the main paper) with different $\lambda_c$. Fig. 2(right) shows the corresponding accuracy versus $T_{in}$. The horizontal lines show the number of inference iterations and accuracy
for BIN. As we can see: (1) CBIN needs much fewer iterations during testing if $T_{in}$ is large enough to get better estimates of $V_S$ during training. (2) CBIN consistently outperforms BIN in a wide range of $T_{in}$. Results for other $V_S$ (and $V$) are consistent with Fig. 2.

### 3.3 Standard Errors

In this section we provide the standard errors (of three trials of different random seeds) in Table 3, Table 4, and Table 5 corresponding to results in the main paper. As we can see, most differences between BIN/CBIN and baselines in accuracy/RMSE are larger than three times the standard errors.

### 4 Intuition and an Illustrative Example for CBIN

As an illustrative example, assume we want to learn a function $v_2 = f_\theta(v_1)$ (ignoring $X$ for simplicity), given 4 data points $\{v_1^{(i)}, v_2^{(i)}\}_{i=1}^4$. Fig. 3 shows the model $v_2 = f_\theta(v_1)$ in the current epoch and the original 4 data points. In the $T_i$ inner loops of Algorithm 1, $v_1^{(i)}$ will be inferred given $v_2^{(i)}$. For example, given $v_2^{(1)}$, Algorithm 1 (in the main paper)
As mentioned in Sec. of the paper, our model naturally generalizes to arbitrary exponential-family distributions (e.g., \(\hat{\theta}^{(1)}\) will infer \(\frac{\partial L}{\partial v^1}\) and use it as \(\hat{v}^{(1)}\). If \(\hat{v}_1\) is too small, it is more possible to get trapped in local minima.

will infer \(\hat{v}_1^{(1)}\) by iteratively computing \(\frac{\partial L}{\partial v_1}\) and updating \(v_1\), where \(L_c = (f_\theta(v_1) - v_2^{(1)})^2\), as shown in Fig. 3(right).

As we can see, the current model has many local optima when inferring \(\hat{v}_1\). Since the resulting \((\hat{v}_1^{(1)}, v_2^{(1)})\) can be viewed as augmented data points, if \((\hat{v}_1^{(1)}, v_2^{(1)})\) is closer to the current \(v_2 = f_\theta(v_1)\) curve, it can make \(v_2 = f_\theta(v_1)\) smoother after updating \(\theta\) in the current epoch, leading to a loss surface \(L_c\) more friendly to backward inference (for \(v_1\)). For instance, we will have the augmented data points as shown in Fig. 3(left) if all \(\hat{v}_1^{(i)}\) can reach the global minima in the inner loops. Incorporating these points (in black) can make \(v_2 = f_\theta(v_1)\) much smoother after updating \(\theta\).

As mentioned in Eqn. 9 of the paper, the augmented data \(\hat{V}_S_i\) is an approximation instead of the true minimizer. Hence in practice the augmented data points may not be exactly on the \(v_2 = f_\theta(v_1)\) curve as shown in Fig. 3(left). Empirically however, we did not find this to be an issue because \(\hat{V}_S_i\) tends to be close enough to the curve if we have sufficient number of inner loop iterations \(T_{in}\), as shown in Fig. 2(b) of the main paper and in Fig. 1(b).

Besides local optima, it is worth noting that when \(V_S\) has more than 1 element, the gradient-based optimization can also be affected by saddle points [6]. Potentially this problem can also be alleviated by extending BIN to CBIN, because the optimization landscape of the objective \(L\) with respect to \(V_S\) can be improved if \(\hat{V}_S\) (inferred using the inner loops of Algorithm 1 in the paper) can successfully escape the saddle points in certain iterations during training.

5 BIN/CBIN for Other Types of Distributions

As mentioned in Sec. of the paper, our model naturally generalizes to arbitrary exponential-family distributions (e.g., gamma distributions), due to the properties of NPN. In this section we briefly introduce BIN/CBIN with gamma distributions and Poisson distributions. Note that it is also possible to have a hybrid BIN/CBIN, where subnetworks belongs to different types of NPNs (e.g., some are Gaussian NPNs and others are gamma NPNs).

5.1 Gamma BIN/CBIN

Essentially gamma BIN/CBIN would replace the Gaussian NPN used in the main paper with gamma NPN. Specifically, the negative log-likelihood in Sec. of the paper:

\[-\log p(v_n | X, V_{n-1}; \theta_n) = \frac{||\mu_{\theta_n}(X, V_{n-1}) - v_n||^2}{2s_{\theta_n}(X, V_{n-1})} + \frac{1}{2} \log s_{\theta_n}(X, V_{n-1})\]  \hspace{1cm} (27)

becomes

\[-\log p(v_n | X, V_{n-1}; \theta_n) = \log \Gamma(c) - c \log d - (c - 1) \log v_n + d v_n,\]  \hspace{1cm} (28)
where
\[ c = \frac{\mu_{\theta_n}(X, V_{n-1})^2}{\sigma_{\theta_n}(X, V_{n-1})}, \quad d = \frac{\mu_{\theta_n}(X, V_{n-1})}{\sigma_{\theta_n}(X, V_{n-1})}. \]

The computation of the hidden layers in gamma NPN is the same as that in [20].

### 5.2 Poisson BIN/CBIN

Besides gamma distributions, BIN/CBIN can also model counts (e.g., word counts in documents). Similar to gamma BIN/CBIN, Eqn. 27 in Poisson BIN/CBIN becomes:

\[ -\log p(v_n|X, V_{n-1}; \theta_n) = -v_n \log c + c + \log(v_n!), \tag{29} \]

where \( v_n \) is a nonnegative integer and

\[ c = \frac{1}{4} \left( 2\mu_{\theta_n}(X, V_{n-1}) - 1 + \sqrt{(2\mu_{\theta_n}(X, V_{n-1}) - 1)^2 + 8\sigma_{\theta_n}(X, V_{n-1})} \right). \]

In practice we need to relax the nonnegative integer \( v_n \) to be a nonnegative real value and replace \( v_n! \) with \( \Gamma(v_n + 1) \) to enable BP. During inference, the predicted real-valued \( v_n \) is then transformed back to a nonnegative integer in the end.

### 6 More Experiment Details and Hyperparameters

We use the published code of [1, 2] implemented in Torch 7 [5] for experiments on SPEN and eSPEN, while our models BIN/CBIN, ‘Retrain’ (NPN), and other baselines are implemented using PyTorch\(^3\).

#### 6.1 Toy Inference Tasks

For the first toy inference task, models are trained for 100 epochs with a minibatch size of 1. We use Adam with a learning rate of 0.005. For CBIN, we set \( \lambda_c = 1 \), the number of inner loop iterations \( T_{in} = 10 \), and the number of warmup epochs \( T_w = 10 \). For the second toy inference task, we use Adam [12] with a learning rate of 0.01 in the training loop and a learning rate of 0.05 in the inner loop. For CBIN, we set \( \lambda_c = 1 \), the number of inner loop iterations \( T_{in} = 20 \), and the number of warmup epochs \( T_w = 10 \). For simplicity, \( s_{\theta_2}() \) is ignored (set to a constant 1) in both tasks. We use multi-layer perceptrons (MLP) with 2 hidden layers of 64 neurons.

#### 6.2 Experiments on Real-world Datasets

In the experiments, all subnetworks (e.g., \( \theta_1 \) and \( \theta_2 \)) share the same encoder which encodes \( X \) into a 512-dimensional (fixed-length) vector. We use 75% of the dataset for training and the rest for testing. Cross validation is performed to determine the best network structures and hyperparameters (see the Supplement for details). We use NPN subnetworks with one hidden layer of 50 neurons after the encoder. For iterative inference on the test set, we perform the same type of inference on the validation set to decide the number of inference iterations. Note that for the inference in the inner loop (with \( T_{in} \) iterations) of Algorithm 1 (in the main paper), one can dramatically speed up the computation by treating all data points as one single minibatch.

For the experiments on \( \text{SHHS2} \), the fixed-length encodings (256, 128, and 128 dimensions for breathing, EEG, and ECG respectively) produced by three encoders, along with \( V_k \), are concatenated to a \( (512 + k) \)-dimensional vector, followed by a hidden NPN layer of 50 neurons and an output NPN layer that produces the mean and variance of \( v_n \). Note that the encoder part is shared across different NPN subnetworks and that the model is trained in an end-to-end fashion. For both real-world datasets, we use the default ordering of variables in the datasets except that we move forward the variable ‘general health’ of \( \text{SHHS2} \) from the last to the third, for convenience of evaluation when \( V = \{ v_n \}_{n=1}^{3} \). For \( \text{SHHS2} \) we tried different ordering when \( V = \{ v_n \}_{n=1}^{8} \), and the results are very similar and consistent to Table 1 in the paper. For Dermatology, \( v_1, v_2, \) and \( v_3 \) are ‘vacuolisation and damage of basal layer’, ‘saw-tooth appearance of retes’, and ‘elongation of the rete ridges’\(^4\).

\(^3\)https://github.com/pytorch/pytorch

\(^4\)We choose 3 histopathological attributes with the the largest average covariance.
As a baseline in the experiments, we combine SV AE \( \text{which later are shared as inputs by all NPN subnetworks of BIN/CBIN.} \) The network is trained in an end-to-end fashion.

Table 6 shows the neural network architecture for the encoders of EEG \( X_e \) and ECG \( X_c \).

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Stride</th>
<th>Channel In</th>
<th>Channel Middle</th>
<th>Channel Out</th>
<th>Type</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>128</td>
<td>128</td>
<td>128</td>
<td>ResBlock</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>128</td>
<td>64</td>
<td>128</td>
<td>ResBlock</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>128</td>
<td>64</td>
<td>256</td>
<td>ResBlock</td>
<td>1</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>256</td>
<td>-</td>
<td>256</td>
<td>SRU</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>256</td>
<td>128</td>
<td>512</td>
<td>ResBlock</td>
<td>1</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>512</td>
<td>-</td>
<td>512</td>
<td>SRU</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>512</td>
<td>256</td>
<td>512</td>
<td>ResBlock</td>
<td>1</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>512</td>
<td>-</td>
<td>512</td>
<td>SRU</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7: Additional layers for encoders of breathing signals \( X_b \).

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Stride</th>
<th>Channel In</th>
<th>Channel Middle</th>
<th>Channel Out</th>
<th>Type</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>5</td>
<td>64</td>
<td>-</td>
<td>64</td>
<td>Conv</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>64</td>
<td>32</td>
<td>64</td>
<td>ResBlock</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>64</td>
<td>32</td>
<td>64</td>
<td>ResBlock</td>
<td>1</td>
</tr>
</tbody>
</table>

For the experiments on Dermatology, we use 80% of the data for training and the rest for testing. Cross validation is performed to decide hyperparameters. Since \( X \) in Dermatology is low-dimensional features, Gaussian NPNs with one hidden layer of 50 neurons are used as subnetworks and no shared encoder is needed. As preprocessing, all attributes are normalized into \([0, 1]\).

In the experiments, the number of warmup epochs \( T_w = 1 \) and the number of inner loop iterations \( T_{in} = 150 \). We use Adam with a learning rate of \(1 \times e^{-4}\) during training and \(1 \times e^{-3}\) during inference. The minibatch size is set to 4. For fairness in the number of free parameters, we also try a larger number of hidden neurons (50 \(\sim\) 400) in the hidden layer of ‘Retrain’, SVAE, DNADE, SPEN, and eSPEN, and the best performance is used in the tables of the main papers (50 is the best choice most of the time). As mentioned in the paper, we assume Gaussian distributions for the variables in both datasets. For the classification task in SHHS2, we use a threshold of 0.5 to process the predicted values.

### 6.3 Network Architecture for Encoders

Table 6 shows the neural network architecture for the encoders of EEG \( X_e \) and ECG \( X_c \). Since the breathing signal \( X_b \) is 10 times the length of \( X_e \) and \( X_c \), additional layers (as shown in Table 7) are needed to align \( X_b \) with \( X_e \) and \( X_c \). We use 1D convolution since \( X \) is time series with multiple channels. ‘ResBlock’ refers to the ResNet block as in our recurrent neural network (RNN) components. ‘Number’ in the tables indicates the number of corresponding blocks stacked in the network. The output of the last SRU will then go through a self-attention layer [16] to output a fix-length encoding, which later are shared as inputs by all NPN subnetworks of BIN/CBIN. The network is trained in an end-to-end fashion.

### 6.4 Details on the SVAE Baseline

As a baseline in the experiments, we combine SVAE [11, 13] and our method to enable BP-based inference and avoid \(O(2^N)\) networks. Specifically, we train the model by maximizing

\[
\log p(V | X) = \log \int p_{\text{enc}}(z | X) p_{\text{dec}}(V | z) dz \\
\geq \int p_{\text{enc}}(z | X) \log p_{\text{dec}}(V | z) dz \\
= E_{p_{\text{enc}}(X, z)} \left[ \log p_{\text{dec}}(V | z) \right],
\]

where \( z \) is the latent variable, \( p_{\text{enc}}(z | X) \) can be seen as the prior on \( z \) using \( X \) as input, and \( V = V_S \cup V_S \) is the set of all variables. Similar to [11, 13], \( p_{\text{enc}}(z | X) \) and \( p_{\text{dec}}(V | z) \) can be learned using BP and the reparameterization trick. Note that since we have the prior \( p_{\text{enc}}(z | X) \) on \( z \), the recognition model \( q(z | V) \) is not needed here.

During the inference phase, using MAP and BP as in BIN, one can infer any subset \( V_S \) from \( V \) and \( V_S \) by finding

\[
\arg\max_{V_S, z} p_{\text{enc}}(z | X) p_{\text{dec}}(V_S, V_S | z).
\]
where \( p_{\text{enc}}(z|X) \) provides regularization when updating \( z \) during inference, and \( p_{\text{dec}}(V_S, V_{-S}|z) \) provides the main gradient, similar to the inferential procedure of BIN.

Note that a learned inferential procedure \( q(z|X, V_{-S}) \) would be subset-specific, meaning \( O(2^N) \) models of \( q(z|X, V_{-S}) \) for \( O(2^N) \) possible subsets; so a feasible method would have to involve an MAP inference over \( z \) as well as \( V_S \), as shown above. This "augmentation" step within MAP can introduce looseness, however. In fact, the same idea is used in sampling methods precisely to avoid getting stuck in local rigid configurations. In contrast, the Bayesian network formulation in BIN captures even rigid interactions very directly.

7 Configuration of \( V_{S_j} \)

As mentioned in the paper, for CBIN, one challenge is that there are \( 2^N - 1 \) configurations of \( V_{S_j} \), including all \( 2^N - 1 \) terms of \( \mathcal{L}_j \) during training is obviously impractical. In our experiments, we let \( J = N - 1 \) and \( V_{S_j} = V_j = \{v_n\}_{n=1}^{2^j} \).

Doing this has the effect of both self-correction and improving the optimization landscape: (1) Self-correction: For example, when \( N = 3 \) and \( j = 2 \), we have \( V_{S_2} = \{\bar{v}_1, \bar{v}_2\} \) and \( V_{S_1} = \{v_3\} \). Since (a) \((\bar{v}_1, \bar{v}_2)\) is different from \((v_1, v_2)\), and (b) \((\bar{v}_1, \bar{v}_2)\) is the current best estimate for \((v_1, v_2)\) given the true \( v_3 \), using \((\bar{v}_1, \bar{v}_2, v_3)\) as "augmented data" to train the \( N = 3 \) subnetworks \((p(v_1|X), p(v_2|X, v_1), \) and \( p(v_3|X, v_1, v_2)\)) in CBIN has the effect of guiding the subnetworks to perform self-correction collaboratively. Here the inner loop inferring \((\bar{v}_1, \bar{v}_2)\) can be seen as searching for the best path for self-correction. (2) Optimization landscape: The effect of improving the optimization landscape comes from using inner loops to infer \( V_j \) given \( V \setminus V_j \) and using \((\bar{V}_j, V \setminus V_j)\) during training. Note that the generated \( \hat{V}_j \) is used as input for \( N - j \) subnetworks (e.g., subnetworks \( p(v_2|X, v_1) \) and \( p(v_3|X, v_1, v_2) \)) both use \( \hat{v}_1 \) as input). Hence using \( N - 1 \) extra terms is sufficient to cover \( N \) subnetworks.

8 Alternative Building Blocks and Related Work

It may be tempting to use probabilistic NN such as \([14, 9]\) as building blocks. However: (1) These methods are designed for binary variables (and assume Bernoulli distributions) while NPN \([20]\) can handle different kinds of variables (e.g., binary variables or continuous variables) and arbitrary exponential-family distributions. (2) More importantly, they do not output the prediction variance, which is crucial to naturally trade off the influence of the prior terms and the terms that provide the main gradient as mentioned in Sec. of the main paper. Besides the choice about building blocks, note that \([14, 9]\) model the joint distributions of binary variables in a deterministic way (models such as \([19]\) can be seen as extension of \([14, 9]\) for real values). Hence they can only predict variables using a feedforward pass and cannot perform inference for the values of an arbitrary set of variables. The best \([14, 9, 19]\) can do is to predict \( V \setminus V_k \) given \( X \) and \( V_k \), which covers only \( N - 1 \) of the \( 2^N - 1 \) cases.

Note that BIN and NADE-based models (including the orderless and real-valued version \([18, 17]\)) are substantially different: (1) BIN performs inference mainly with backpropagation, while NADEs perform inference the usual way with feedforward. (2) NADEs use parameter sharing to parameterize different conditionals with a single network while BIN parameterizes each conditional with an NPN; (3) BIN supports arbitrary Bayesian Network structures while NADEs do not; this allows convenient incorporation of domain knowledge; (4) BIN can be extended to CBIN to improve accuracy/efficiency. Finally, the large performance gap in Table 2~4 empirically verifies these differences.

9 Figures in the Paper

In this section we provide additional illustrative figures in Fig. 4 (an example inference process of BIN) and larger versions of some figures in the main paper for readers’ convenience in Fig. 5.
Figure 4: Illustration of the inference process of BIN when $V_S = \{v_1\}$ and $V_{-S} = \{v_2\}$. FF provides prior (regularization) for $v_1$ and BP provides the main gradients to update $v_1$. Circles and rectangles in grey indicates observed variables and fixed networks parameters, respectively.
Step 1: Iterative FF/BP to infer $v_1$

Step 2: One-pass FF to predict $v_3$

NPN 1
NPN 2
NPN 3

Step 1: Iterative FF/BP to infer $v_1$

Step 2: One-pass FF to predict $v_3$

$V_S = \{ v_1, v_3 \}$ and $V_{-S} = \{ v_2 \}$. Edges in different colors correspond to different NPN subnetworks. Best viewed in color. Middle: An example for forward prediction of a more general BN structure. Bottom: An example for hybrid inference of a more general BN structure.

Figure 5: An example for hybrid inference when $V_S = \{ v_1, v_3 \}$ and $V_{-S} = \{ v_2 \}$. Edges in different colors correspond to different NPN subnetworks. Best viewed in color. Middle: An example for forward prediction of a more general BN structure. Bottom: An example for hybrid inference of a more general BN structure.
References


