Approximate Inference（近似推論）

Chapter 19

楊　森（Sen Yang）2019/08/18

Inferenceは一部変数が与えられて、残る部分変数の確率分布を計算すること（つまり条件付き分布を計算すること）。この中に、潜在変数を含む確率モデルをトレーニングする場合、p(h|v)の計算は中心課題です。

Challenge of Inference（推論）

$p(h \mid v)$の計算もしくは $p(h \mid v)$ 分布での期待値の計算

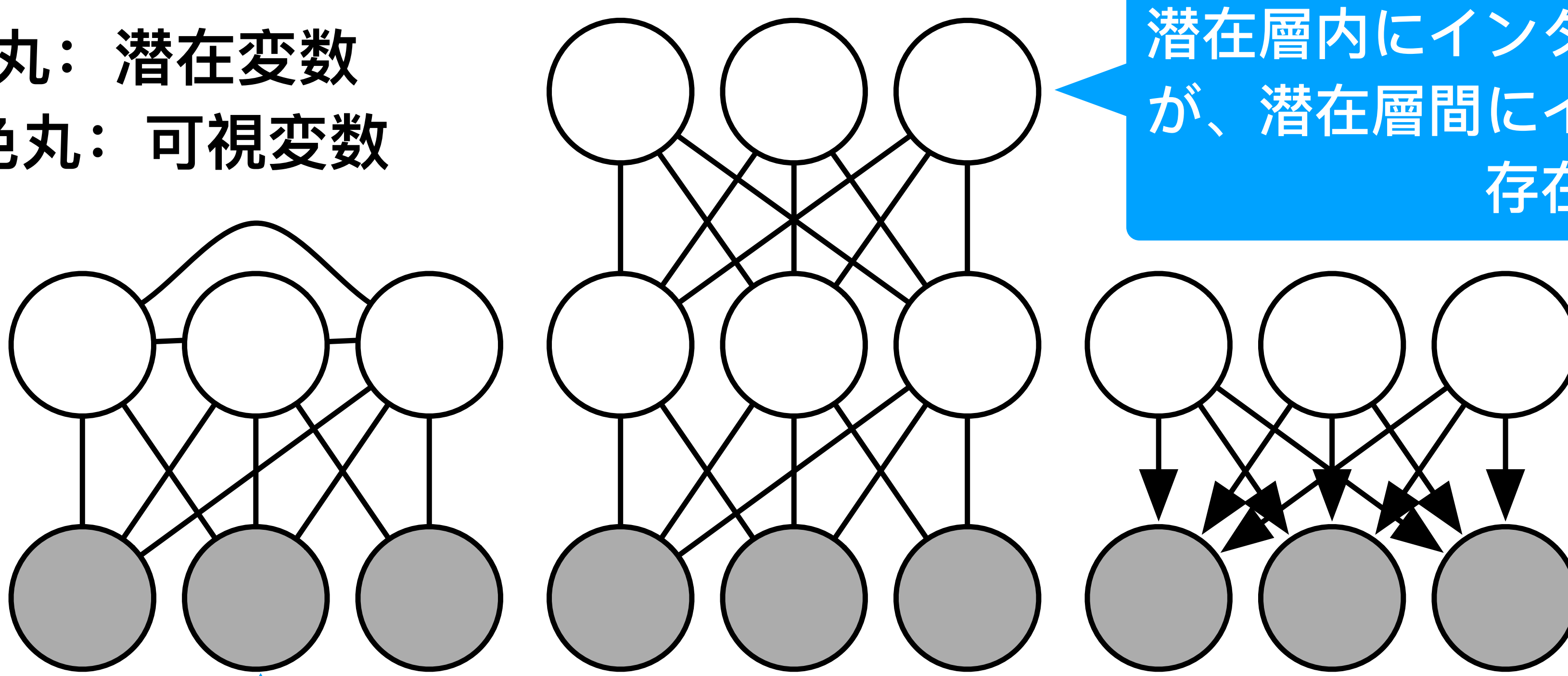
潜在変数
latent
variable

可視変数
visible
variable

この計算はmaximum likelihood learning（最尤学習）に必要

Inferenceが解けない場合の理由：潜在変数間のインタラクションが存在する

白丸：潜在変数
灰色丸：可視変数



潜在層内にインタラクションがないが、潜在層間にインタラクションが存在する

子ノードが見えたら、親ノード間インタラクションが発生する

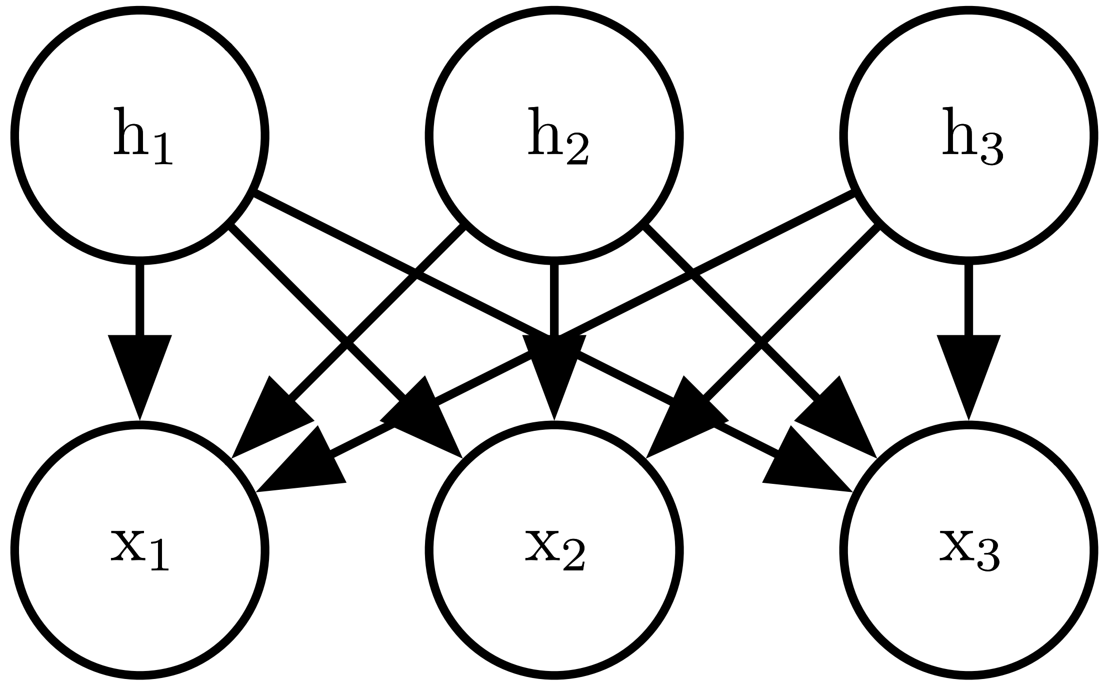
semi-restricted Boltzmann machine (undirected)

deep Boltzmann machine (undirected)

directed model

Chapter 16 Structured Probabilistic Modelの内容

参考：Linear Factor Models (Chapter 13)



$$\mathbf{x} = \boldsymbol{W}\mathbf{h} + \boldsymbol{b} + \mathbf{noise}$$

Linear factor modelは潜在変数の分布p(h)を階乗分布（factorial distribution）にしている

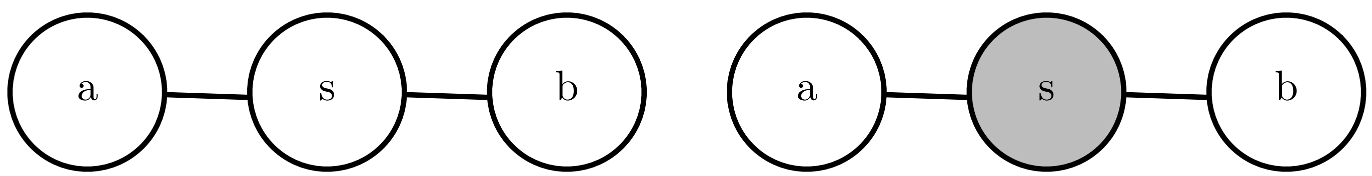
$$p(h) = \prod_i p(h_i)$$

つまり潜在変数がお互いに独立しているのを仮定する

・factorial分布だと、inferenceも簡単だし、samplingも簡単
・factorial分布仮定は後ほど出る（Variational Inference and Learningでの mean field方法）

参考：Separation (undirected models) and D-Separation (directed models) (Chapter 16 16.2.5)

Separationの意味は、conditional independence（条件付独立）

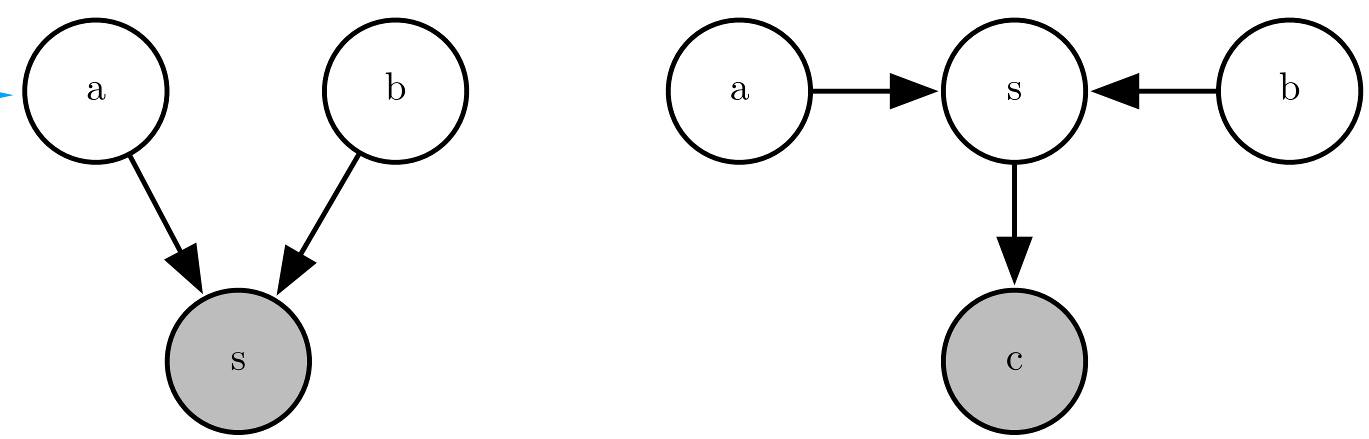


(a) (b)

sが見えたら、aとbのパスは非アクティブ。（aとbがseparate）
sが見えなかったら、aとbのパスはアクティブ。（aとb not separate）

directed modelの特別な依存関係：子ノードもしくは子ノードの後継が見えたら、親ノード間に依存がある（explain away effect）

例えば、s: 同僚が出社していない
a: 同僚が病気
b: 同僚が旅行中
もしsが見えたら、aとbは依存関係が出そう。例えば、sが見える場合、aが発生したら、bの可能性が非常に低くなる。



(c) (d)

Inference as Optimization

最尤学習では $logp(v;\theta)$ を計算すること。

そのため、$p(v,h;\theta)$ でvの周辺分布を計算すれば良いが、コストが高い

代わりに、$logp(v;\theta)$ の下界 $\mathscr{L}(v,\theta,q)$（ELBO, evidence lower bound）を計算

$$\mathscr{L}(v,\theta,q) = logp(v;\theta) - D_{KL}(q(h \mid v) \| p(h \mid v;\theta))$$

qは任意のhの分布。Dは常に非負数。qとpが同じ分布の場合、ELBOイコール
logp(v;θ)。qがpに近づけば近づくほどELBOがタイトになる。

（ELBO, evidence lower bound）の良く使う形：（変換は次のスライド）

$$\mathscr{L}(v,\theta,q) = \mathbb{E}_{h \sim q}[logp(h,v)] + H(q)$$

ELBOのOptimizationで良いqを出せる、つまりp(h|v)の良いInferenceを出せる

Inference as Optimization: ELBOを組み替える

$$
\begin{aligned}
\mathscr{L}(v,\theta,q) &= logp(v;\theta) - \mathbb{E}_{h\sim q} log \frac{q(h|v)}{p(h|v)} \\
&= logp(v;\theta) - \mathbb{E}_{h\sim q} log \frac{q(h|v)}{\frac{p(h,v;\theta)}{p(v;\theta)}} \\
&= logp(v;\theta) - \mathbb{E}_{h\sim q}[logq(h|v) - logp(h,v;\theta) + logp(v;\theta)] \\
&= -\mathbb{E}_{h\sim q}[logq(h|v) - logp(h,v;\theta)] \\
&= \mathbb{E}_{h\sim q}[logp(h,v;\theta)] + H(q)
\end{aligned}
$$

VAE（Variational Autoencoder）の学習もELBOに基づく。次のスライドでまず1つ目ELBOを最適化する方法EM（Expectation Maximization）を紹介する。

Expectation Maximization (EM)

$$\mathscr{L}(v,\theta,q) = -\mathbb{E}_{h\sim q}[logq(h\,|\,v) - logp(h,v;\theta)]$$

$$= \mathbb{E}_{h\sim q}[logp(h,v;\theta)] + H(q)$$

MステップでMonte Carloサンプリングで期待値を計算する為にqのサンプルを用意する。（自分の理解）
Monte Carlo Samplingは次のスライド

収束までEステップとMステップを繰り返す。

E-step：qの更新

$$q(h^{(i)}\,|\,v) = p(h^{(i)}\,|\,v^{(i)};\theta^{(0)})$$

$v^{(i)}$ はi番目trainingサンプル　$\theta^{(0)}$はステップ開始時のパラメータ

M-step：θの更新

$$\sum_i \mathscr{L}(v^{(i)},\theta,q)$$ をθに対して最大化にする

M-stepでθが変わるとqも変わるはずだが、E-stepで取ったqサンプルを使う。

E-stepでq=p（exact inference）なので、ELBO=log p(v;θ)。M-stepでθが変わって、q変わらないので、ELBOとlog p(v;θ)の差がどんどん出る。またE-stepに入ると、差が0になる。EMは近似推論q(h|v)だけじゃなく、近似事後（posterior）分布p(h|v)を出せる。

参考：Basics of Monte Carlo Sampling (Chapter 17 17.1.2)

厳密に計算できないsumもしくは積分（integral, continued）を近似する方法
例えばある分布での期待値を計算する時（分布がわからなくても、サンプリングできれば）

$$s = \sum_x p(x)f(x) = E_p[f(x)] \qquad s = \int p(x)f(x) = E_p[f(x)]$$

n個サンプルを取って平均値を近似値として使う

$$\hat{s}_n = \frac{1}{n}\sum_{i=1}^{n} f(x^{(i)})$$

このestimatorはunbiasedだ：

$$\mathbb{E}[\hat{s}_n] = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[f(x^{(i)})] = \frac{1}{n}\sum_{i=1}^{n} s = s$$

このestimatorのvarianceは：

$$Var[\hat{s}_n] = \frac{1}{n^2}\sum_{i=1}^{n} Var[f(x)] = \frac{Var[f(x)]}{n}$$

このbasic Monte CarloサンプリングでELBOにある期待値を全部計算できる

MAP（最大事後確率） Inference and Sparse Coding

最大事後確率推定と最尤推定の区別は次のスライドへ

考え方：最尤q(h|v)ではなく、最尤h（点推定）を推定して、ELBOを最適化する

$$\mathscr{L}(v, \theta, q) = \mathbb{E}_{h\sim q}[logp(h, v)] + H(q)$$

①MAP推定問題になるように、qをディラック分布に限定

$$H(q) = -\mathbb{E}_{h\sim q}logq(h|v)$$

$$q(h|v) = \delta(h-\mu) = \begin{cases} +\infty, & h = \mu \\ 0, & h \neq \mu \end{cases}$$

$$= -\int \delta(x)log\delta(x)dx$$

H(q)が定数なので、省略可

②ELBOに代入して、最尤h推定問題は、MAP推定問題になる

$$\mu^* = \underset{\mu}{argmax}\, logp(h=\mu, v)$$

$$h^* = \underset{h}{argmax}\, logp(h, v)$$

$$= \underset{h}{argmax}\, log\frac{p(h, v)}{p(v)}$$

$$= \underset{h}{argmax}\, logp(h|v)$$

潜在変数hのMAP推定問題

③EMと同じように、E-stepにMAP推定でh*を推定；M-stepにθを更新して、logp(h*,v)を最適化する

参考：MAP（Maximum A Posteriori, 最大事後確率）推定とML（Maximum Likelihood, 最尤）推定（Chapter 5 5.5, 5.6）

最尤推定：

$$
\begin{aligned}
\theta_{ML} &= \underset{\theta}{argmax}\, p_{model}(\mathbb{X};\theta) \\
&= \underset{\theta}{argmax} \prod_{i=1}^{m} p_{model}(x^{(i)};\theta) \\
&= \underset{\theta}{argmax} \sum_{i=1}^{m} log p_{model}(x^{(i)};\theta) \\
&= \underset{\theta}{argmax}\, \mathbb{E}_{x\sim\hat{p}_{data}} log p_{model}(x;\theta)
\end{aligned}
$$

最尤推定で、θは確率変数ではない。θは未知だが固定。（頻度論統計学の考え方、frequentist statistics）

最大事後確率推定で、θは見えないので、確率変数だ。（ベイズ統計学の考え方、Bayesian statistics）

最大事後確率推定：

$$
\theta_{MAP} = \underset{\theta}{argmax}\, p(\theta\,|\,x) = \underset{\theta}{argmax}\, log p(x\,|\,\theta) + log p(\theta)
$$

log尤度

θの事前確率分布を使う

MAP（最大事後確率） Inference and Sparse Coding (Chapter 13.4)

Deep Learningの中に、MAP推定は主にSparse Codingモデルに使われている

①Sparse CodingもLinear Factorモデルの一つ、hの事前確率分布はラプラス分布：

$$p(h_i) = \frac{\lambda}{2} e^{-\lambda |h_i|}$$

②vは線形Decoder＋ノイズで生成する：

$$p(v \mid h) = \mathcal{N}(v; Wh + b, \frac{1}{\beta} I)$$

③ページ2,4紹介した子ノード(v)が見えたら親ノード(h)間依存あり、独立ではない原因でp(h|v)は解けないことで、MAP推定で、最尤hだけを推定する：

$$h* = \underset{h}{argmax}\, p(h \mid v) = \underset{h}{argmax}\, logp(h \mid v) = \underset{h}{argmax}\, logp(v \mid h)$$

$$+\underset{h}{argmax}\, logp(h) = \underset{h}{argmin}\, \lambda \|h\|_1 + \beta \|v - Wh\|_2^2$$

④training setのhをHに、vをVに入れて、Sparse Codingモデルの最適化目標関数：

$$J(H, W) = \sum_{i,j} |H_{i,j}| + \sum_{i,j} (V - HW^T)^2_{i,j}$$

Sparse Codingモデルのトレーニングは Jを最小化する

Variational（変分） Inference and Learning

考え方：　$\mathbb{E}_q logp(h, v)$を簡単に計算できるように、q(h|v)を制限する。良く使われる制限は、factorial分布：

$$q(h \mid v) = \prod_i q(h_i \mid v)$$

mean field（平均場、平均ば）方法。変分方法の綺麗さは、qを制限する時、何のパラメータも使わないこと。

連続潜在変数の場合：変分法（calculus of variations）で関数スペースから一つの分布関数を決める。変分法紹介は後ほど。

変分法はVariational Inference名前の由来が、離散の場合変分法はいらない

離散潜在変数の場合：普通にELBOを最適化できる

例えば、バイナリSparse Codingモデルの場合

$$q(h_i = 1 \mid v) = \hat{h}_i$$

$$\mathscr{L}(v, \theta, \hat{h}) = \mathbb{E}_{h \sim q}[logp(h, v)] + H(q)$$

$$p(h_i = 1) = \sigma(b_i)$$

$$p(v \mid h) = \mathcal{N}(v; Wh, \beta^{-1})$$

収束まで更新。$\hat{h}$を同時に更新できない。収束標準は、Lが上がらない、もしくは$\hat{h}$が変わらない

$$\frac{\partial}{\partial \hat{h}_i} \mathscr{L}(v, \theta, \hat{h}) = 0$$

この不動点（fixed-point）更新推定ルールを適用後の式次スライドへ

Variational（変分） Inference and Learning

$$\frac{\partial}{\partial \hat{h}_i}\mathscr{L}(v,\theta,\hat{h}) = 0$$ を展開後、RNNになる：

$$\hat{h}_i = \sigma(b_i + v^T \beta W_{:,i} - \frac{1}{2} W_{:,i}^T \beta W_{:,i} - \sum_{j \neq i} W_{:,j}^T \beta W_{:,i} \hat{h}_j)$$

インプット

この式はRNNなので、直接にトレーニングしても良い

組み替えて：

$$\hat{h}_i = \sigma(b_i + (v - \sum_{j \neq i} W_{:,j}\hat{h}_j)^T \beta W_{:,i} - \frac{1}{2} W_{:,i}^T \beta W_{:,i})$$

隠れユニットiは、他の隠れユニットを使って、インプット(v)をEncodeしている

離散の場合、実践者は自分で変分法を解く必要がなく、mean fieldで良く使われるq(h|v)不動点更新式がある：

$$\tilde{q}(h_i \mid v) = exp(\mathbb{E}_{h_{-i} \sim q(h_{-i} \mid v)} log \tilde{p}(v, h))$$

この期待値を計算すると、q(hi|v)の関数式が出る。

例（p640-641）は略。

参考：変分法（Calculus of Variations, Chapter 19 19.4.2）

Functional（関数の関数）： J[f]

Functional derivatives / Variational derivatives : $\frac{\delta}{\delta f(x)} J$

Functional derivativeの一つのプロパティ（式①）：

$$\frac{\delta}{\delta f(x)} \int g(f(x), x) dx = \frac{\partial}{\partial y} g(f(x), x) \Rightarrow \frac{\partial}{\partial \theta_i} \sum_j g(\theta_j, j) = \frac{\partial}{\partial \theta_i} g(\theta_i, i)$$

左式の直感的な理解：θi以外の微分は0なので、θiでの微分だけ残る

分布関数のエントロピー（Functional）：

$$H[p] = -\mathbb{E}_x logp(x) = -\int p(x) logp(x) dx$$

Functional微分でH[p]が最大時のpを求める：しかしpが分布にならないかもしれないので、制限が必要。例えば、p(x)の積分が1になること、平均値、分散。ラグランジュの未定乗数法（Lagrange multiplier）でこれらの制限を入れる。新しいpのFunctionalは：

$$\mathscr{L}[p] = \lambda_1\left(\int p(x)dx - 1\right) + \lambda_2(\mathbb{E}[x] - u) + \lambda_3(\mathbb{E}[(x-\mu)^2] - \sigma^2) + H[p]$$

$$= \int (\lambda_1 p(x) + \lambda_2 p(x)x + \lambda_3 p(x)(x-\mu)^2 - p(x)logp(x))dx - \lambda_1 - \mu\lambda_2 - \sigma^2\lambda_3$$

参考：変分法（Calculus of Variations, Chapter 19 19.4.2）続

前ページのラグランジュを最小化(?)するため、Functional微分を0にする：

$$\frac{\delta}{\delta p(x)}\mathscr{L} = \lambda_1 + \lambda_2 x + \lambda_3 (x-\mu)^2 - 1 - log p(x) = 0$$

前ページの変分法の式①を使う

組み替えて：

$$p(x) = exp(\lambda_1 + \lambda_2 x + \lambda_3 (x-\mu)^2 - 1)$$

制限を満たすために、λを設定して、p(x)が正規分布になる：

$$\lambda_1 = 1 - log\sigma\sqrt{2\pi}, \lambda_2 = 0, \lambda_3 = -\frac{1}{2\sigma^2} \Rightarrow p(x) = \mathcal{N}(x; \mu, \sigma^2)$$

だから分布がわからない時、正規分布を使う。正規分布の時のエントロピーが一番大きい。

Learned Approximate Inference（学んだ近似推定）

考え方：ELBOの最適化プロセスを下記のマッピングと見る

$$v \xrightarrow{\quad f \quad} q^* = \underset{q}{argmax}\, \mathscr{L}(v, q)$$

Neural network（推定ネットワーク）をトレーニングして、fに近似する

①Wake-Sleep：p(v|h)でhからvまでサンプリングして、(v, h)をトレーニングデータとして推定ネットワークをトレーニングする。

②学んだ近似推定はVariational Autoencoderに使われて、生成モデルの主な手法の一つになっている。
推定ネットワークはターゲットがなし、ただELBOの一部として定義されて、ネットワークのパラメータはELBOを最大化するために調整される。