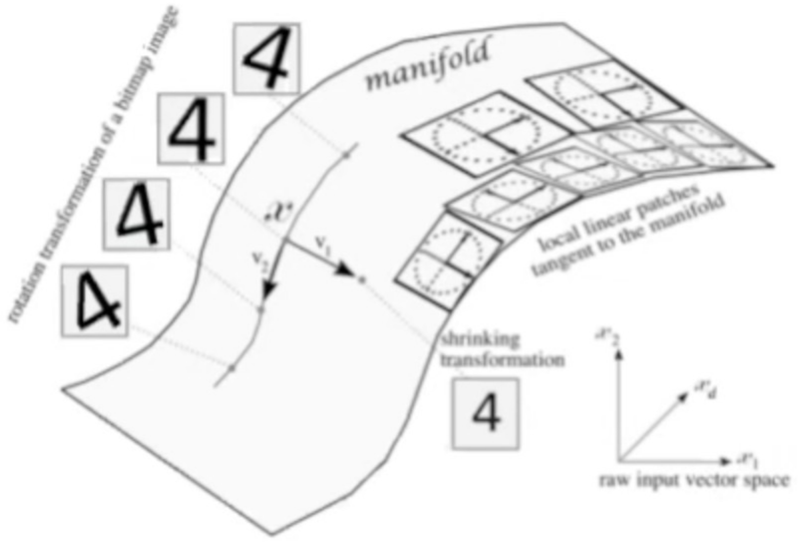
Stochastic Backpropagation And Approximate Inference in Deep Generative Models (2014)

Introduction

Manifold hypothesis: real world high-dimensional data lie on low-dimensional manifolds embedded within the high-dimensional space

Importance in Deep Learning: This hypothesis indicates that a deep learning algorithm only needs to learn to focus on few key features of dataset. However these key features may turn out to be complicated functions of the original variables(latent variables) => need to find out relationship


rotation transformation of a bitmap image
manifold
x
v_1
v_2
local linear patches tangent to the manifold
shrinking transformation
x_1
x_2
x_d
raw input vector space

Introduction

Traditional latent variable models are easy to sample, but efficiency has been in question

Seek for:

(1) Deep (Complex) Structured models

(2) Allow for fast sampling

(3) Computationally tractable and scalable to high-dimensional data

- Tractability: Allow integrals to be computed

- Scalability: Adapt well even if dimension or dataset has changed

Introduction

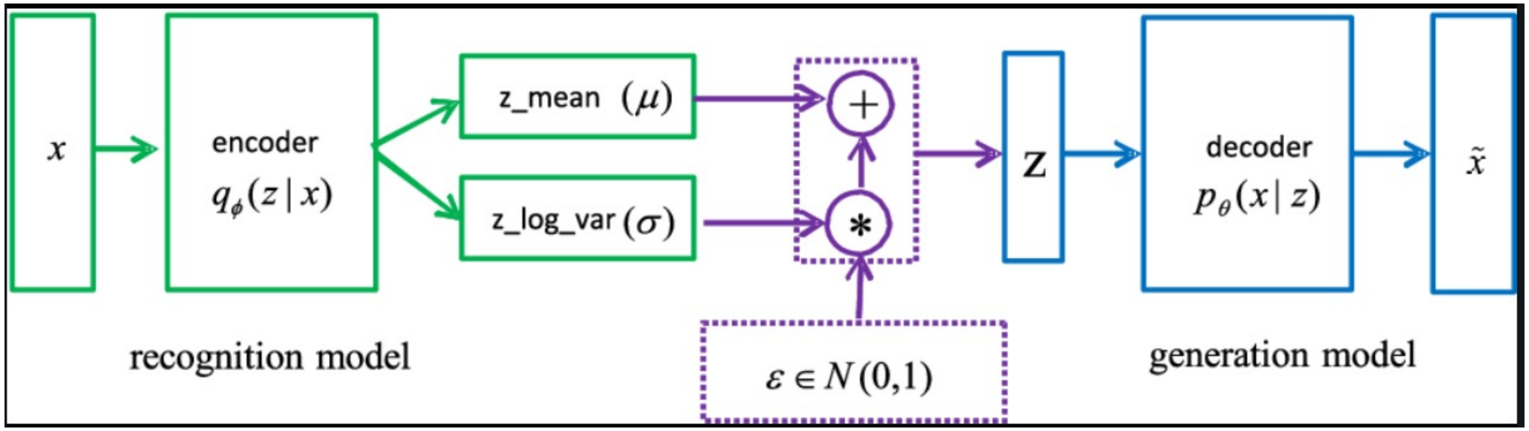
(1) => introduce deep, directed generative models with Gaussian latent variables at each layer

(2) => optimization of parameters of generative and recognition models jointly

(3) => introduce approximate representation of the posterior over the latent variables using a recognition model

generative model -> derive objective function using variational principles

recognition model -> specify structure utilizing deep learning



Deep Latent Gaussian Models(DLGM))

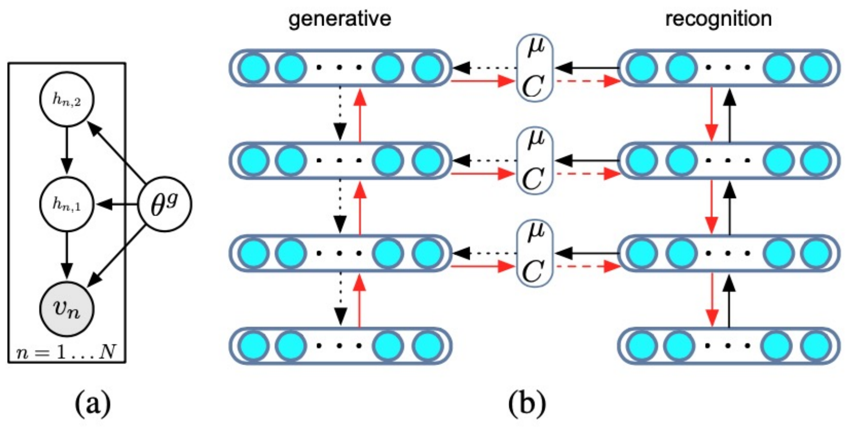


Figure 1. (a) Graphical model for DLGMs (5). (b) The corresponding computational graph. Black arrows indicate the forward pass of sampling from the recognition and generative models: Solid lines indicate propagation of deterministic activations, dotted lines indicate propagation of samples. Red arrows indicate the backward pass for gradient computation: Solid lines indicate paths where deterministic backpropagation is used, dashed arrows indicate stochastic backpropagation.

<Fig 1-(a)>

- consists L layers of latent variables
- begin at the top most layer by drawing from Gaussian distribution
- activation h_l at any lower layer is formed by non-linear transformation of above layer h_{l+1}, perturbed by Gaussian noise
- generate observations v by sampling from the observation likelihood using the activation of the lowest layer h_1

$$\boldsymbol{\xi}_l \sim \mathcal{N}(\boldsymbol{\xi}_l|\mathbf{0},\mathbf{I}), \quad l = 1,\dots,L$$
$$\mathbf{h}_L = \mathbf{G}_L\boldsymbol{\xi}_L,$$
$$\mathbf{h}_l = T_l(\mathbf{h}_{l+1}) + \mathbf{G}_l\boldsymbol{\xi}_l, \quad l = 1\dots L-1$$
$$\mathbf{v} \sim \pi(\mathbf{v}|T_0(\mathbf{h}_1)),$$

- θ^g is the set of parameters that include T_l and G_l
- $\pi(v|.)$ is obtained by approximation
- adopt a gaussian prior $p(\theta^g) = N(\theta|0, KI)$

DLGM

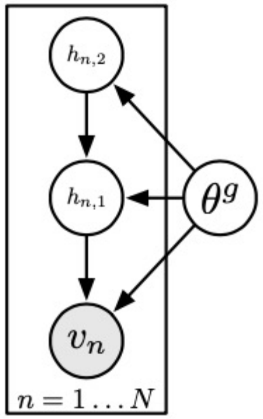
Then the joint probability can be expressed in two equivalent ways

$$p(\mathbf{v},\mathbf{h}) = p(\mathbf{v}|\mathbf{h}_1,\boldsymbol{\theta}^g)p(\mathbf{h}_L|\boldsymbol{\theta}^g)p(\boldsymbol{\theta}^g)\prod_{l=1}^{L-1} p_l(\mathbf{h}_l|\mathbf{h}_{l+1},\boldsymbol{\theta}^g)$$

$$p(\mathbf{v},\boldsymbol{\xi}) = p(\mathbf{v}|\mathbf{h}_1(\boldsymbol{\xi}_{1\ldots L}),\boldsymbol{\theta}^g)p(\boldsymbol{\theta}^g)\prod_{l=1}^{L}\mathcal{N}(\boldsymbol{\xi}|\mathbf{0},\mathbf{I}).$$

(proof omitted)

the second line implies that the transformed distribution tries to match the empirical distribution $p(\boldsymbol{\xi}) = \prod_{l=1}^{L}\mathcal{N}(\boldsymbol{\xi}_l|\mathbf{0},\mathbf{I})$

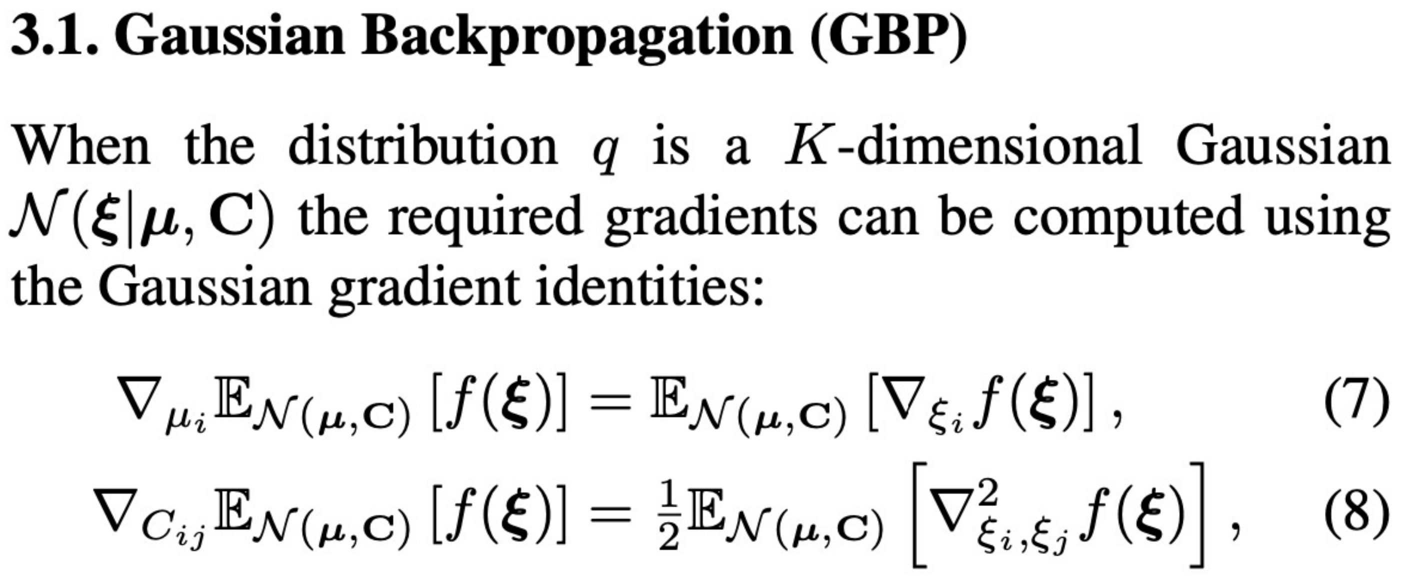


(a)

Stochastic Backpropagation

Usually the gradient descent requires computation of $\nabla_\theta \mathbb{E}_{q_\theta}\left[f(\boldsymbol{\xi})\right]$

But it is difficult to compute when
(i) expectation is unknown
(ii) indirect dependency on the parameters of q over which the expetation is taken (expectation 하고도, parameter에 depend)



3.1. Gaussian Backpropagation (GBP)

When the distribution q is a K-dimensional Gaussian $\mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\mu}, \mathbf{C})$ the required gradients can be computed using the Gaussian gradient identities:

$$\nabla_{\mu_i} \mathbb{E}_{\mathcal{N}(\boldsymbol{\mu},\mathbf{C})}\left[f(\boldsymbol{\xi})\right] = \mathbb{E}_{\mathcal{N}(\boldsymbol{\mu},\mathbf{C})}\left[\nabla_{\xi_i} f(\boldsymbol{\xi})\right], \tag{7}$$

$$\nabla_{C_{ij}} \mathbb{E}_{\mathcal{N}(\boldsymbol{\mu},\mathbf{C})}\left[f(\boldsymbol{\xi})\right] = \frac{1}{2}\mathbb{E}_{\mathcal{N}(\boldsymbol{\mu},\mathbf{C})}\left[\nabla^2_{\xi_i,\xi_j} f(\boldsymbol{\xi})\right], \tag{8}$$

Stochastic Backpropagation

※ expectation 이 아닌, cost function 안에 gradient가 들어가기에, 위의 두 가지 문제점 없어짐

Assuming that we can combine two gradients above (parameters are related),

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathcal{N}(\boldsymbol{\mu},\mathbf{C})}\left[f(\boldsymbol{\xi})\right] = \mathbb{E}_{\mathcal{N}(\boldsymbol{\mu},\mathbf{C})}\left[\mathbf{g}^{\top}\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\theta}} + \frac{1}{2}\mathrm{Tr}\left(\mathbf{H}\frac{\partial \mathbf{C}}{\partial \boldsymbol{\theta}}\right)\right] \qquad (9)$$

where g and H are the gradient and the Hessian of the function f() respectively

Stochastic Backpropagation

For non-Gaussian case:

1. Using the product rule for integrals : find a function $B(\boldsymbol{\xi};\boldsymbol{\theta})$ to ensure that

$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{p(\boldsymbol{\xi}|\boldsymbol{\theta})}[f(\boldsymbol{\xi})] = -\mathbb{E}_{p(\boldsymbol{\xi}|\boldsymbol{\theta})}[\nabla_{\boldsymbol{\xi}}[B(\boldsymbol{\xi};\boldsymbol{\theta})f(\boldsymbol{\xi})]].$$

=> are used in Gaussian, inverse Gamma and log-Normal

2. Using suitable coordinate transformations

=> For a case when a distribution can be written as smooth, invertible transformation of a standard base distribution

ex) any Gaussian $N(\mu,\sigma^2)$ can be obtained as a transformation of Gaussian $N(0,1)$ using a transformation $y=\mu+\sigma*\epsilon$

$$\nabla_R E_{N(\mu,\sigma^2)}[f(\xi)] = \nabla_R E_{N(0,1)}[f(\mu+\sigma\epsilon)]$$

Scalable Inference in DLGM

- To make bayesian inference, we need to marginalize out latent variables and compute posterior distribution

- This requires to compute the integrated or marginal likelihood

- In general, this will be intractable and we will instead optimize lower bound on the marginal likelihood

$$\begin{aligned}\mathcal{L}(\mathbf{V}) &= -\log p(\mathbf{V}) = -\log \int p(\mathbf{V}|\boldsymbol{\xi},\boldsymbol{\theta}^g)p(\boldsymbol{\xi},\boldsymbol{\theta}^g)d\boldsymbol{\xi} \\ &= -\log \int \frac{q(\boldsymbol{\xi})}{q(\boldsymbol{\xi})} p(\mathbf{V}|\boldsymbol{\xi},\boldsymbol{\theta}^g)p(\boldsymbol{\xi},\boldsymbol{\theta}^g)d\boldsymbol{\xi} \qquad (11)\end{aligned}$$

$$\leq \mathcal{F}(\mathbf{V}) = \underbrace{D_{KL}[q(\boldsymbol{\xi})\|p(\boldsymbol{\xi})]}_{\text{KL-divergence}} - \underbrace{\mathbb{E}_q\left[\log p(\mathbf{V}|\boldsymbol{\xi},\boldsymbol{\theta}^g)p(\boldsymbol{\theta}^g)\right]}_{\text{Reconstruction Error}}. \qquad \text{: Evidence L.B}$$

Scalable Inference

- approximate posterior distribution $q(\xi|v)$, which is conditioned on the observed data is usually obtained by solving KL-divergence

- for simplicity, we use $q(\xi|v)$ that is a Gaussian distribution across L layers:

$$q(\boldsymbol{\xi}|\mathbf{V},\boldsymbol{\theta}^r) = \prod_{n=1}^{N}\prod_{l=1}^{L}\mathcal{N}\left(\boldsymbol{\xi}_{n,l}|\boldsymbol{\mu}_l(\mathbf{v}_n),\mathbf{C}_l(\mathbf{v}_n)\right), \quad (12)$$

where the mean $\boldsymbol{\mu}_l(\cdot)$ and covariance $\mathbf{C}_l(\cdot)$ are generic maps represented by deep neural networks. Parameters of the q-distribution are denoted by the vector $\boldsymbol{\theta}^r$.

- For a Gaussian prior and Gaussian recognition model, the KL-term can be computed analytically and our objective function becomes

$$D_{KL}[\mathcal{N}(\boldsymbol{\mu},\mathbf{C})\|\mathcal{N}(\mathbf{0},\mathbf{I})] = \tfrac{1}{2}\left[\mathrm{Tr}(\mathbf{C}) - \log|\mathbf{C}| + \boldsymbol{\mu}^\top\boldsymbol{\mu} - D\right],$$

$$\mathcal{F}(\mathbf{V}) = -\sum_{n}\mathbb{E}_q\left[\log p(\mathbf{v}_n|\mathbf{h}(\boldsymbol{\xi}_n))\right] + \tfrac{1}{2\kappa}\|\boldsymbol{\theta}^g\|^2$$

$|C| = \det(c)$

$$+\frac{1}{2}\sum_{n,l}\left[\|\boldsymbol{\mu}_{n,l}\|^2 + \mathrm{Tr}(\mathbf{C}_{n,l}) - \log|\mathbf{C}_{n,l}| - 1\right], \quad (13)$$

Scalable Inference

$$D_{KL}[\mathcal{N}(\boldsymbol{\mu},\mathbf{C})\|\mathcal{N}(\mathbf{0},\mathbf{I})] = \tfrac{1}{2}\left[\mathrm{Tr}(\mathbf{C}) - \log|\mathbf{C}| + \boldsymbol{\mu}^\top\boldsymbol{\mu} - D\right],$$

$$\mathcal{F}(\mathbf{V}) = \boxed{-\sum_n \mathbb{E}_q\left[\log p(\mathbf{v}_n|\mathbf{h}(\boldsymbol{\xi}_n))\right] + \tfrac{1}{2\kappa}\|\boldsymbol{\theta}^g\|^2} + \frac{1}{2}\sum_{n,l}\left[\|\boldsymbol{\mu}_{n,l}\|^2 + \mathrm{Tr}(\mathbf{C}_{n,l}) - \log|\mathbf{C}_{n,l}| - 1\right], \quad (13)$$

- Define distribution of q(.) to be a recognition model

- To optimize the previous objective function, we use Monte Carlo methods for any expectations and use stochastic gradient descent for optimization

- We require efficient estimators of the gradients of all terms in equation w.r.t θ^g (generative model parameters) and θ^r (recognition model parameters)

- The gradients w.r.t the jth generative parameter can be computed using stochastic gradient descent:

$$\nabla_{\theta_j^g}\mathcal{F}(\mathbf{V}) = -\mathbb{E}_q\left[\nabla_{\theta_j^g}\log p(\mathbf{V}|\mathbf{h})\right] + \tfrac{1}{\kappa}\theta_j^g.$$

Scalable Inference

$$D_{KL}[\mathcal{N}(\boldsymbol{\mu},\mathbf{C})\|\mathcal{N}(\mathbf{0},\mathbf{I})] = \tfrac{1}{2}\left[\mathrm{Tr}(\mathbf{C}) - \log|\mathbf{C}| + \boldsymbol{\mu}^\top\boldsymbol{\mu} - D\right],$$

$$\mathcal{F}(\mathbf{V}) = \boxed{-\sum_n \mathbb{E}_q\left[\log p(\mathbf{v}_n|\mathbf{h}(\boldsymbol{\xi}_n))\right]} + \tfrac{1}{2\kappa}\|\boldsymbol{\theta}^g\|^2$$
$$+\frac{1}{2}\sum_{n,l}\left[\|\boldsymbol{\mu}_{n,l}\|^2 + \mathrm{Tr}(\mathbf{C}_{n,l}) - \log|\mathbf{C}_{n,l}| - 1\right], \quad (13)$$

- To obtain gradients w.r.t recognition parameter θ^r, we use Gaussian backpropagation

- To address the difficulty of computing Hessian, we use co-ordinate to write $C = RR^T$

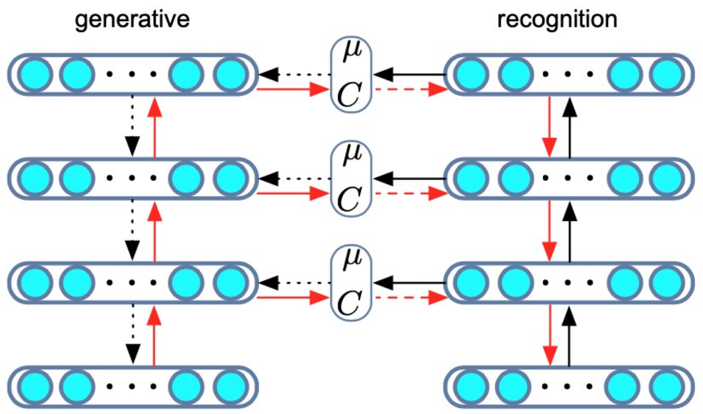
- Then we can obtain the following gradients (proof omitted):

$$\nabla_{\boldsymbol{\theta}^r}\mathcal{F}(\mathbf{v}) = \nabla_{\boldsymbol{\mu}}\mathcal{F}(\mathbf{v})^\top \frac{\partial\boldsymbol{\mu}}{\partial\boldsymbol{\theta}^r} + \mathrm{Tr}\left(\nabla_{\mathbf{R}}\mathcal{F}(\mathbf{v})\frac{\partial\mathbf{R}}{\partial\boldsymbol{\theta}^r}\right). \quad (17)$$

Scalable Inference

Algorithm 1 Learning in DLGMs

```
while hasNotConverged() do
    $\mathbf{V} \leftarrow$ getMiniBatch()
    $\boldsymbol{\xi}_n \sim q(\xi_n|\mathbf{v}_n)$ (bottom-up pass) eq. (12)
    $\mathbf{h} \leftarrow \mathbf{h}(\xi)$ (top-down pass) eq. (3)
    updateGradients() eqs (14) – (17)
    $\boldsymbol{\theta}^{g,r} \leftarrow \boldsymbol{\theta}^{g,r} + \Delta\boldsymbol{\theta}^{g,r}$
end while
```



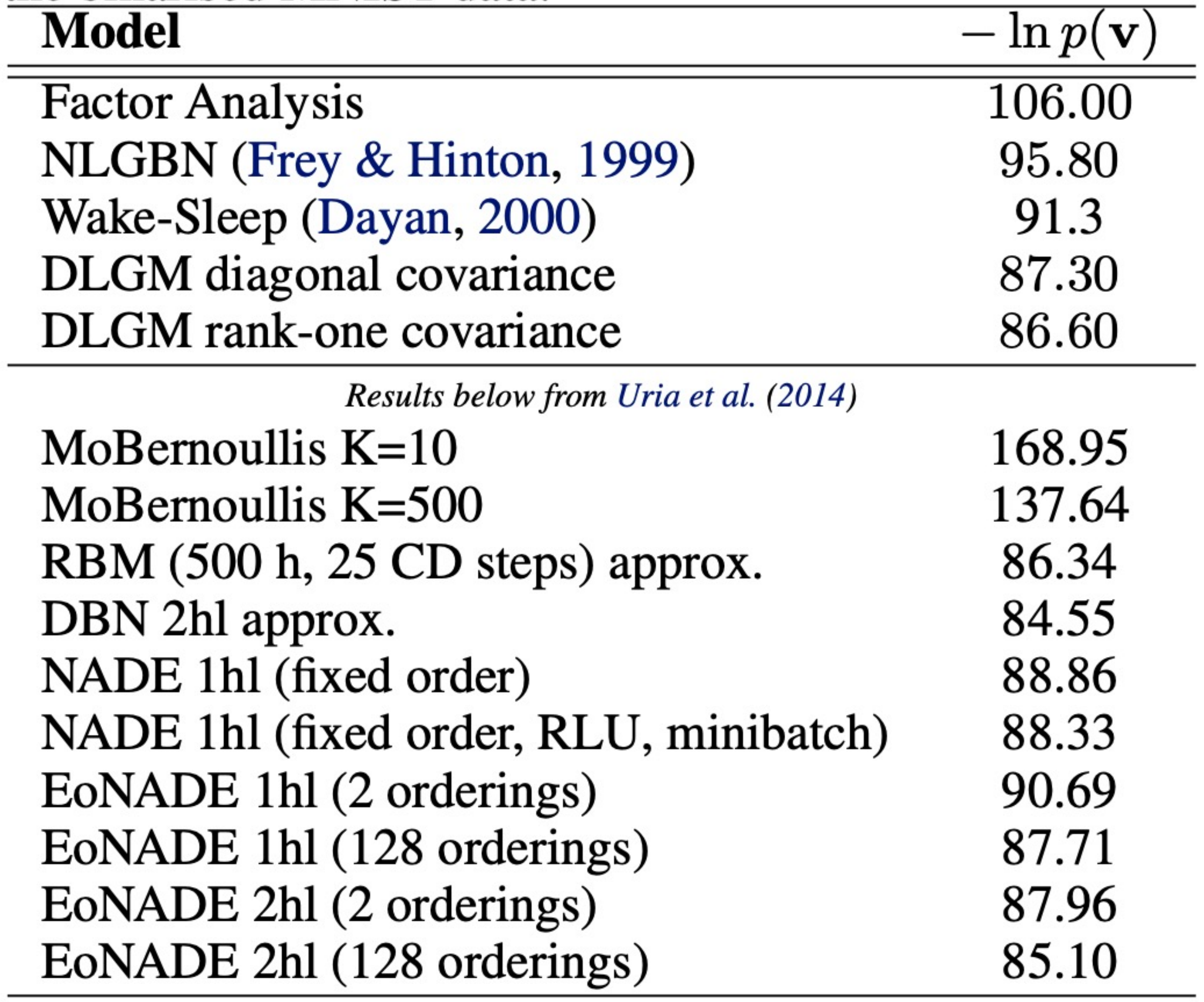
(b)

1. Forward pass (black arrows) consisting of bottom-up(recognition) and top-down(generation) phase => update hidden activations and parameters of Gaussian distributions

2. Backward pass (red arrows) which gradients are computed using deterministic/stochastic backpropagation

Results

Table 1. Comparison of negative log-probabilities on the test set for the binarised MNIST data.



Model	$-\ln p(\mathbf{v})$
Factor Analysis	106.00
NLGBN (Frey & Hinton, 1999)	95.80
Wake-Sleep (Dayan, 2000)	91.3
DLGM diagonal covariance	87.30
DLGM rank-one covariance	86.60
Results below from Uria et al. (2014)	
MoBernoullis K=10	168.95
MoBernoullis K=500	137.64
RBM (500 h, 25 CD steps) approx.	86.34
DBN 2hl approx.	84.55
NADE 1hl (fixed order)	88.86
NADE 1hl (fixed order, RLU, minibatch)	88.33
EoNADE 1hl (2 orderings)	90.69
EoNADE 1hl (128 orderings)	87.71
EoNADE 2hl (2 orderings)	87.96
EoNADE 2hl (128 orderings)	85.10

Conclusion

처음에 했던 문제제기들

1. Complex Structure을 잘 잡아줄 수 있냐

2. Sampling을 정확하고 빠르게 할 수 있는가

3. Tractable and Scalable 한가

1. Using Deep Latent Gaussian Models (L latent layer)

2. Stochastic Gradient Descent를 통해 정확하게 수렴시킬 수 있고, 다른 베이지안 추론방법인 Mean-Field Approximation이나 EM-algorithm보다 posterior를 generalization하는 것이 빠른 convergence와 inference 보장

3. Using Variational Inference (approximating a posterior)

Contribution to VAE

- Recognition Model (Encoder) 과 Generative Model(Decoder)을 동시에 학습시켜 줌으로써 불필요한 낭비 제거

- Variational Inference에 Stochastic Gradient Descent를 적용시켜 모델의 정확성을 높였음