# Is Anyone interest in auto-encoding variational-Bayes ? instead of "VAE"

2018. 9. 21.

Mingyu Kim

# Contents

1. TBD

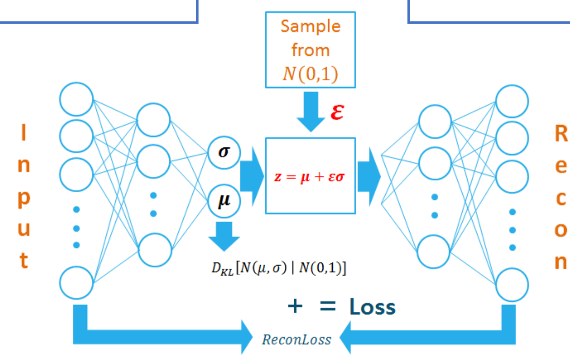
# INTRODUCTION TO VAE (I)

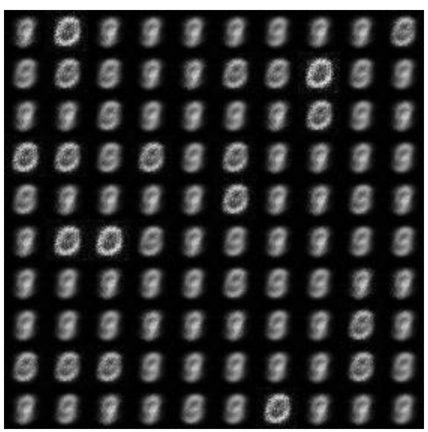
**Loss Function**

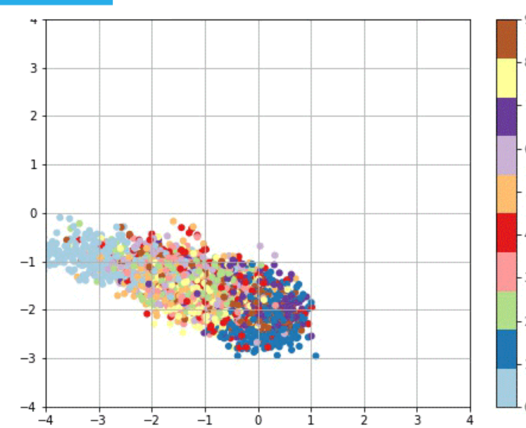
- Variational Inference

**Neural network architecture**

- Auto encoder
- Mixture Density Network





# Introduction to VAE (2)

- **Characteristics**
  - **Scalable** generative model
    - **Amortized** variational inference
    - Simple network architecture
    - Stochastic gradient descent (Back-propagation)
      - Scales to huge datasets
  - **Inference** based on probability graphical model
    - Continuous Latent Space
    - Data reduction / Data Imputation
      - Liable to noisy

# This Seminar

- **Paper "Auto-encoding variational bayes"**
  - "VAE" is a example of experiment results in this paper.
- **Introduce the rest of this paper without "VAE"**
  - Stochastic Gradient Variational Bayes(SGVB) estimator
    - Auto-encoding Variational Bayes algorithm
  - Reparameterization trick
  - Appendix
    - Full Variational Bayes
    - Marginal Likelihood estimator
- **Research trend of VAE**

# *PRELIMINARY*

TRADITIONAL VARIATIONAL INFERENCE

# Bayesian Inference

- **General expression**
  - A random variable $x \in X$
  - A set $\{y_1, \cdots, y_N\}$ of i.i.d observations

$$p(x, y_1, \cdots, y_N) = p_0(x) \prod_{\{n=1\}}^{N} p(y_n|x)$$

  - Given prior $p_{0(x)}$ and observations, posterior can be evaluated.

$$p(x|y_1, \cdots, y_N) = \frac{p_0(x) \prod_{\{n=1\}}^{N} p(y_n|x)}{Z} \coloneqq \frac{\bar{p}(x)}{Z}$$

$$Z \coloneqq \int_X \bar{p}(x) dx$$

# Variational Inference

- **Variational Inference**
  - Approximate the target distribution $p(x)$ by using a tractable distribution $q(x)$ in Q

$$q^* = argmin_{q \in Q} KL(q||p)$$

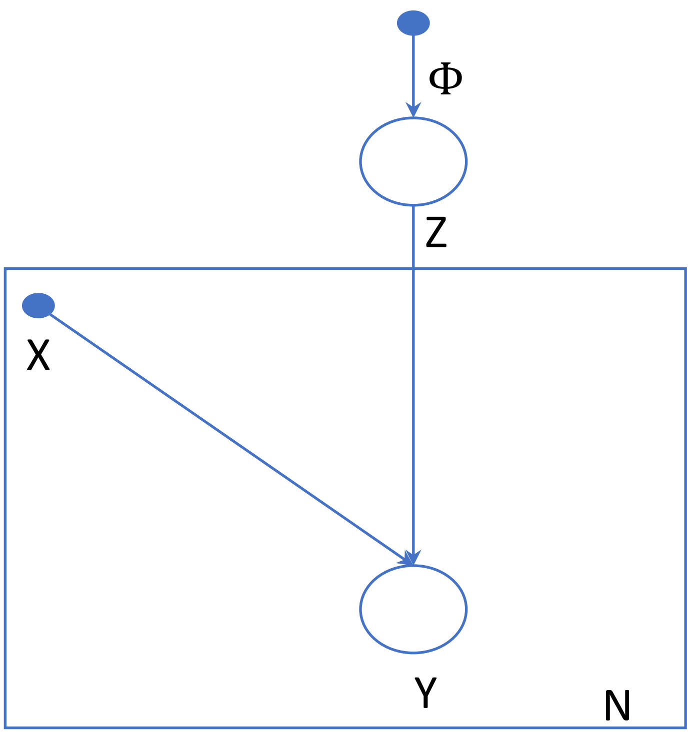
- **Variational Bound**

$$logp(x) = \int logp(x) \cdot q(z)dz = \int log \frac{p(x,z)}{p(z|x)} \cdot q(z)d$$

$$= \int log \frac{p(x,z)}{q(z)} \cdot q(z)dz - \int log \frac{q(z)}{p(z|x)} \cdot q(z)dz$$

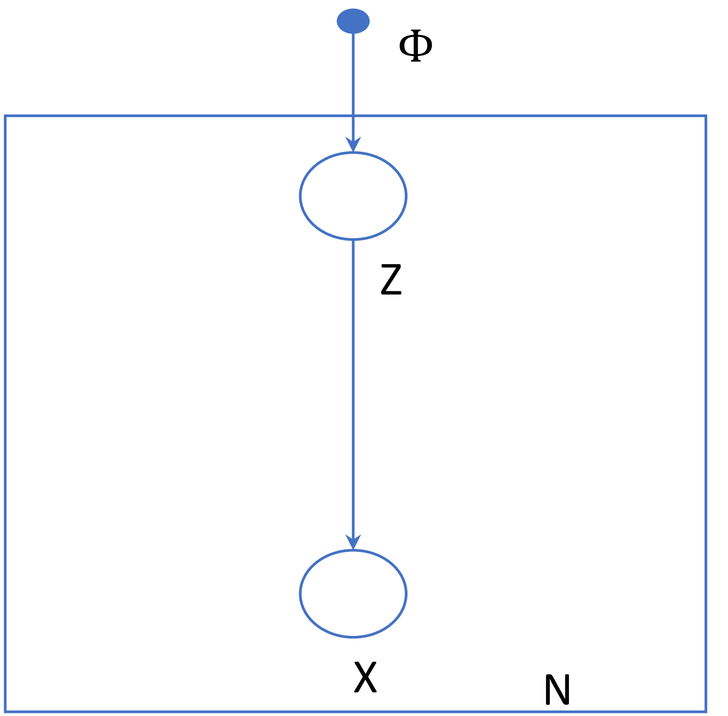
# Probability Graphical Model

- **Supervised Learning**




e.g) Bayesian Logistic regression

- **Unsupervised Learning**
(Latent Variable Model)




e.g) EM algorithm for
Gaussian Mixture Model

# Probability Graphical Model

- **Supervised learning**

  e.g) Bayesian logistic regression

$$p(x|w,y)\cdot p(w) \propto p(w,x|y)$$

$p(x|w,y)$ : $\prod_{\{i=1\}}^{n} \sigma(w^T x)^{y_i} \cdot (1-\sigma(w^T x)^{1-y_i})$ → Approximate Gaussian Dist.

- **Latent Variable Model**

  e.g) Gaussian Mixture Model (EM algorithm)

  → tractable conditional pdf : $p(z|x)$

$$q(z) \approx p\left(z\middle|x,\theta^{old}\right)$$

$$\theta^* = argmax_\theta \sum_z p\left(z\middle|x,\theta^{old}\right)\cdot lnp(x,z|\theta)$$

# Traditional Variational Inference

- **Mean field assumption**
  - If $p(z|x)$ is intractable ?

$$q(\boldsymbol{z}) = \prod_i q_i(z_i)$$

$$lnq_j^*(z_j) = E_{i\neq j}[lnp(\boldsymbol{x},\boldsymbol{z})] + const$$

$$q_j^*(z_j) = \frac{\exp(E_{i\neq j}[lnp(\boldsymbol{x},\boldsymbol{z})]}{\int \exp(E_{i\neq j}[lnp(\boldsymbol{x},\boldsymbol{z})]dz}$$

  - Under specific probabilistic graphical model, approximate distribution can be evaluated by expectation of $E_{i\neq j}[lnp(\boldsymbol{x},\boldsymbol{z})]$
  - Depending on each problem, we have to derive the form of $E_{i\neq j}[lnp(\boldsymbol{x},\boldsymbol{z})]$
  - Through sequential optimization, All of $q_j(z_j)$ can be updated in order until converged.

# Traditional Variational Inference (2)

- **VI with Stochastic gradient descent**

$$L = \int log \frac{p(x,z)}{q_\phi(z)} \cdot q_\phi(z)dz$$

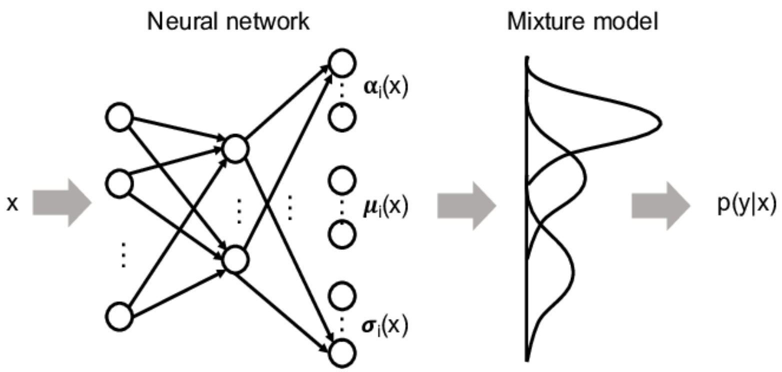
$$\nabla_\phi L = \int lnp(x,z) \cdot \nabla_\phi lnq(z) \cdot q(z)dz = E_{x\sim q}[lnp(x,z) \cdot \nabla_\phi lnq(z)]$$

$$\nabla_\phi L = E_{x\sim q}\left[lnp(x,z) \cdot \nabla_\phi lnq(z)\right] \approx \frac{1}{n}\sum_i lnp(x_i, z\,) \cdot \nabla_\phi lnq(z)$$

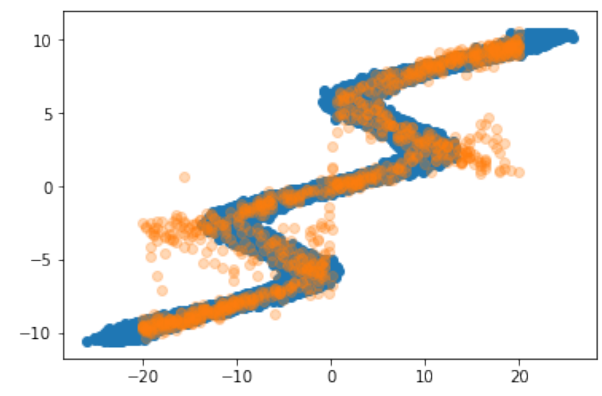
- In order to avoid exact derivation of expectation and sequential optimization, stochastic gradient descent method was introduced.
- This method only requires $logp(x,z),\ q_\phi(z)$ and its derivative
- All components can be easily handed
    - $logp(x,z)$ can be derived from probabilistic graphical model
    - $q_\phi(z)$ and $\nabla_\phi q_\phi(z)$ are decided by users.
- However, since this estimator has a huge variance, it also requires many samples.

# Network for learning parameters

- **Mixture density network**




[ Structure of Mixture density Network ]




[ Data point and result by Mixture Density Network]

$$E(w) = -\sum_{\{n=1\}}^{N} \ln\{\sum_{\{k=1\}}^{K} \pi_k(x_n, w) N(t_n | u_k(x_n, w), \sigma_k^2(x_n, w))\}$$

Mixture density network presents that artificial neural network trains hyper-parameters for learning parameters of distributions

# Chapter Summary

- **Variational Inference**
  - It provide the evidence in which mathematical analysis can be conducted
  - Before emerging amortized VI, Most researches had a interest in efficient usage of VI using stochastic gradient descent and improving performance of approximate distribution for classification model

    : Basic example : Bayesian logistic regression.
  - Especially, VI with stochastic gradient descent influenced that “Auto-encoding variational bayes” were published.

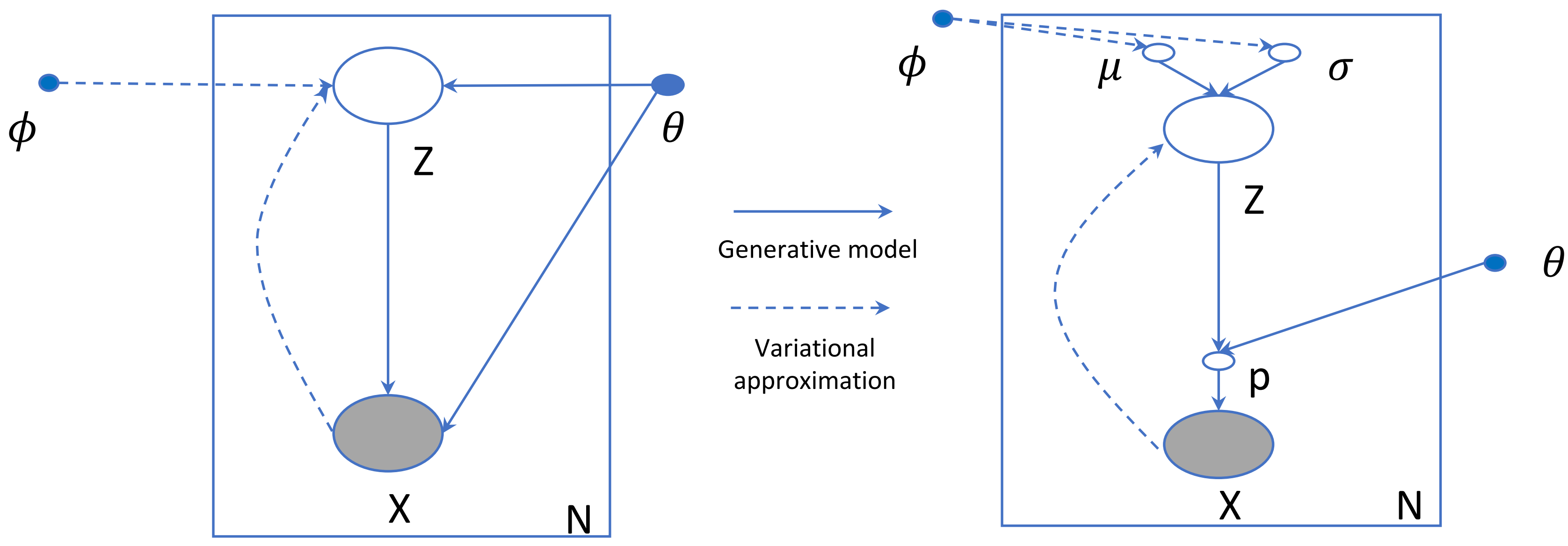
- **Mixture Density Network**
  - Mixture Density Network is known to learn parameters of specific distributions using NN
  - It is starting point to derive an approach to bayesian neural network

# AUTO-ENCODING VARIATIONAL BAYES

PROBABILISTIC GRAPHICAL MODEL
STOCHASTIC VARIATIONAL GRADIENT BAYES ESTIMATOR

# Probabilistic graphical model

- **Amortized generative model**




[ AEVB algorithm ]

[ VAE ]

- $\phi$ : hyper-parameters in recognition network
- $\theta$ : hyper-parameters in generative network

# Problems Scenario

- **Traditional Latent variable model**
  - Intractable : $p(z|x)$
  - A large data set : Sampling based EM algorithm → Too slow

- **This paper has interested in**
  - Using stochastic gradient descent
  - Efficient approximate posterior inference of latent variable $z$

- **This paper introduce** (The reason why this method referred "Variational")
  - $q_{\phi}(z|x)$ : recognition model / approximate posterior distribution *to* $p_{\theta}(z|x)$
    - Figure out parameters of approximate distribution
  - $p_{\theta}(x|z)$ : generative model (defined as graphical model)

  - Learning the recognition model parameter $\phi$ with generative model parameter $\theta$ using reparameterization trick

# Variational Bound

- Developing variational Bound

$$\int log \frac{p(x,z)}{q(z)} \cdot q(z)dz = \int log \frac{p(x|z) \cdot p(z)}{q(z)} \cdot q(z)dz$$

$$\int \left(logp(x|z) + logp(z) - logq(z)\right) \cdot q(z)dz$$

$$= \int logp(x|z) \cdot q(z)dz + \int logp(z) - logq(z)dz$$

$$= \int logp(x|z) \cdot q(z)dz - D_{KL}(q(z)||p(z)$$

# Stochastic Gradient Variational Bayes Estimator (1)

- **Reparametrization Trick**
  - Random variable z → Function of variables (Deterministic)
    - Capable of using neural network
    - $z \sim q(z|x)$ → $\bar{z} = g_{\phi}(x, \epsilon)$, $\epsilon \sim p(\epsilon)$
      - Assume that $p(\epsilon)$ is defined ($p(\epsilon) \sim N(0,1)$)
      - The function $g_{\theta}$ is determined depending on $q(z|x)$
        (In subsequent chapter, the forms of function $g_{\theta}$ is represented)

  - Form of function $g_{\theta}$
    - Tractable inverse CDF : similar to finding PDF
      - $F(x) = p, F^{-1}(p) = x \ (unique\ x)$ / $g_{\phi}(\epsilon, x) = \mathrm{F}_{\phi}^{-1}(\epsilon; x) = \bar{Z}$
    - Location Scale Model :
      - $g_{\phi}(\epsilon, x)$ = location + scale $\cdot\ \epsilon$

# Stochastic Gradient Variational Bayes Estimator (2)

- **SGVB Estimator**

$$E_{q_{\phi_{(z|x^i)}}}[f(z)] = E_{p(\epsilon)}\left[f\left(g_\phi(\epsilon, x^i)\right)\right] \approx \frac{1}{L}\sum_i f\left(g_\phi(\epsilon^l, x^i)\right)$$

$$where, \epsilon^l \sim p(\epsilon)$$

  - $f(z) = logp(x|z) + q(z) - p(z)$
    - $Given\ x, \epsilon\ sampled\ from\ p(\epsilon), z \approx \bar{z} = g_\phi(\epsilon, x^i), f(z)$ can be evaluated.
  - The form of $\frac{1}{L}\sum_i f\left(g_\phi(\epsilon^l, x^i)\right)$
    - Stochastic gradient method can be introduced w.r.t $\phi$ , $\theta$

      (Back propagation)

# Stochastic Gradient Variational Bayes Estimator (3)

- **SGVB Estimator A**

$$\widetilde{L^A}(\theta,\phi;x^i)=\frac{1}{L}\sum_l logp_\theta\left(x^i,z^{i,l}\right)-logq_\phi(z^{i,l}|x^i)$$

$$where, g_\phi\left(\epsilon,x^i\right),\epsilon^l\sim p(\epsilon)$$

  - $logp_\theta\left(x^i,z^{i,l}\right)$ is defined by probabilistic graphical model
  - $logq_\phi(z^{i,l}|x^i)$ can be determined $g_\phi\left(\epsilon,x^i\right),\epsilon^l\sim p(\epsilon)$

- **SGVB Estimator B**

$$\widetilde{L^B}(\theta,\phi;x^i)=\frac{1}{L}\sum_l logp_\theta\left(x^i|z^{i,l}\right)-D_{KL}(q_\phi(z|x^i)||p_\theta(z))$$

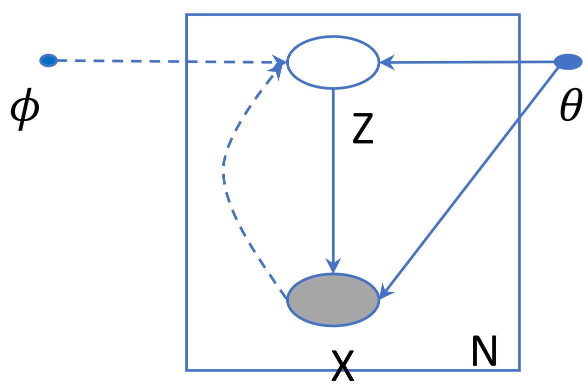
$$where, g_\phi\left(\epsilon,x^i\right),\epsilon^l\sim p(\epsilon)$$

  - $D_{KL}(q_\phi(z|x^i)||p_\theta(z))$ is analytically evaluated (Gaussian Dist.)
  - $D_{KL}(q_\phi(z|x^i)||p_\theta(z))$ doesn't require sample z, only require parameter of approximate dist.

# Auto Encoding Variational bayes Algorithm(1)

**Algorithm 1** Minibatch version of the Auto-Encoding VB (AEVB) algorithm. Either of the two SGVB estimators in section 2.3 can be used. We use settings $M = 100$ and $L = 1$ in experiments.

$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
**repeat**
  $\mathbf{X}^M \leftarrow$ Random minibatch of $M$ datapoints (drawn from full dataset)
  $\boldsymbol{\epsilon} \leftarrow$ Random samples from noise distribution $p(\boldsymbol{\epsilon})$
  $\mathbf{g} \leftarrow \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \widetilde{\mathcal{L}}^M(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{X}^M, \boldsymbol{\epsilon})$ (Gradients of minibatch estimator (8))
  $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Update parameters using gradients $\mathbf{g}$ (e.g. SGD or Adagrad [DHS10])
**until** convergence of parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$
**return** $\boldsymbol{\theta}, \boldsymbol{\phi}$




$$ln p_\theta(X, Z) = \sum_i ln p_\theta(x^i | z^i)$$

$$ln q_\phi(Z) = \sum_i ln q_\phi(z^i)$$

$$ln p_\theta(Z) = \sum_i ln p_\theta(z^i)$$

# AUTO ENCODING VARIATIONAL BAYES ALGORITHM (2)

- **AEVB for SGVB estimator**

$$\frac{1}{M}\sum_{i}\widetilde{L^{A}}(\theta,\phi;x^{i}) \ = \frac{1}{M}\sum_{i}\frac{1}{L}\sum_{l} logp_{\theta}\left(x^{i},z^{i,l}\right)-logq_{\phi}(z^{i,l}|x^{i})$$

$$\frac{1}{M}\sum_{i}\widetilde{L^{B}}(\theta,\phi;x^{i})=\frac{1}{M}\sum_{i}\frac{1}{L}\sum_{l} logp_{\theta}\left(x^{i}|z^{i,l}\right)-D_{KL}(q_{\phi}(z|x^{i})||p_{\theta}(z))$$

# Chapter Summary

- **SGVB Estimator**
  - This estimator was developed to use an approach of neural network for learning probabilistic latent variable model
  - Reparameterization trick is the key point to implement SGVB estimator

- **AEVB algorithm**
  - It can train a neural network using gradient of loss function (SGVB estimator) w.r.t hyper-parameters in recognition and generative network at once.
  - This gradient can be evaluated by back-propagation

# VARIATIONAL AUTO ENCODER

FRAMEWORK
NETWORK ARCHITECTURE
FULL VARIATIONAL BAYES
MARGINAL LIKELIHOOD ESTIMATOR

# FRAMEWORK (1)

- **Loss function** (Bernoulli Case)
  - $p(z) \sim N(0, I)$
  - $p_\theta(x|z) \sim N\big(u_\theta(z), \sigma_\theta(z)\big)\ or\ Bern(\mathrm{p}_\theta(z))$
  - $q_\phi(z|x) \sim N(u_\phi(x), diag_\phi(x))$

- **Estimator (Loss function)**

$$\widetilde{L^B}(\theta, \phi; x^i) = \frac{1}{L}\sum_{l} log p_{\theta}\left(x^i | z^{i,l}\right) - D_{KL}(q_\phi(z|x^i)||p_\theta(z))$$

$$where\ g_\phi\left(\epsilon, x^i\right) = u_\theta(x) + \epsilon \cdot \sigma_\theta(x)$$

$$\epsilon \sim N(0, I), \qquad l = 1$$

  - Since $l = 1$, neural network architecture can be similar to "auto-encoder"

# FRAMEWORK (2)

- **Loss function**
  - $p_\theta(x|z) \sim Bern(\mathrm{p}_\theta(z))$

    $\rightarrow logp_\theta(x|z) \sim \sum_i x_i log\mathrm{p}_\theta(z_i) + (1 - x_i)log(1 - \mathrm{p}_\theta(z_i))$
  - $D_{KL}(q_\phi(z|x^i)||p_\theta(z))$
    - $p(z) \sim N(0, I),\ q_\phi(z|x) \sim N(u_\phi(x), diag_\phi(x))$
    - $D_{KL}(q_\phi(z|x)||p_\theta(z)) =$

$$\frac{1}{2}\{\left(0 - u_\phi(x)\right) I^{-1} \left(0 - u_\phi(x)\right) + trace(I^{-1}diag_\phi(x)) - k + ln\frac{|I|}{|diag_\phi(x)|}$$

$$= \sum_j u_\phi(x)_{(j)}^2 - 1 + \sigma_\phi^2(x)_{(j)} - ln\sigma_\phi^2(x)_{(j)}$$

$where\ j = index\ of\ dimension\ of\ latent\ variable\ z$

# FRAMEWORK (3)

- **Loss Function**

$\therefore \sum_i x_i log\mathrm{p}_{\theta}(z_i) + (1 - x_i) log(1 - \mathrm{p}_{\theta}(z_i))$ +

$$\sum_i \sum_j u_{\phi}(x_i)^2_{(j)} - 1 + \sigma^2_{\phi}(x_i)_{(j)} - ln\sigma^2_{\phi}(x_i)_{(j)}$$

# STRUCTURE OF VAE(1)

- **Network Architecture**
  - Recognition network (like density network)

$$u_\phi(x_i) = w_2 tanh(w_1 x_i + b_1) + b_2$$
$$\sigma_\phi(x_i) = w_4 tanh(w_3 x_i + b_3) + b_4$$

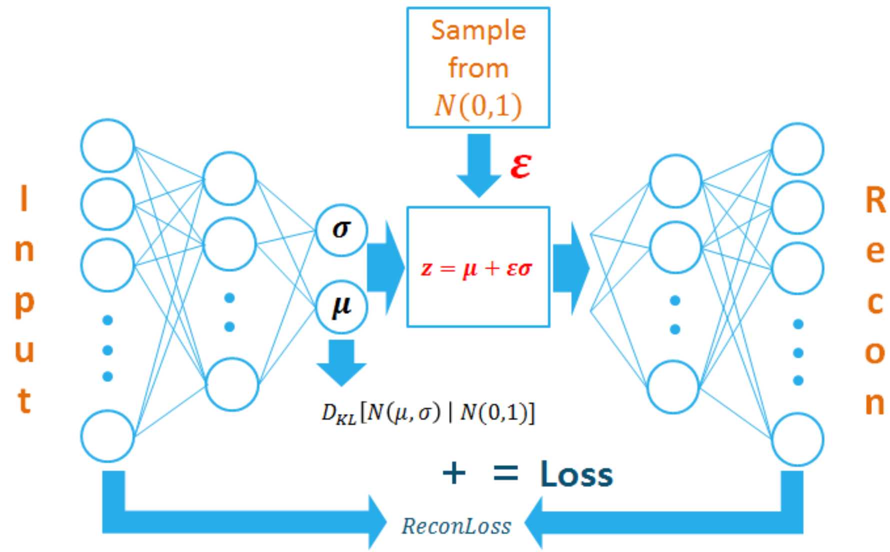
  - Generative network
    - Bernoulli Case

$$\mathrm{p}_\theta(z_i) = f_a(w_2 \tanh(w_1 z_i + b_1) + b_2)$$
$$where\ f_a(\cdot): element - wise\ sigmoid\ function$$
$$\theta = \{w_1, b_1, w_2, b_2\}$$

    - Gaussian Case

$$u_\theta(z_i) = w_4(tanh(w_3 z + b_3) + b_4$$
$$\sigma_\theta(z_i) = w_5(tanh(w_3 z + b_3) + b_5$$
$$\theta = \{w_3, w_4, w_5, b_3, b_4, b_5\}$$

# VAE




- **Probabilistic modeling**
  - All data should be mapped to latent space defined by probability graph model
  - Robust against some noises
    - Key difference between VAE and general AE

# Marginal Likelihood Estimator (1)

- **Derivation**
  - Marginal Likelihood : $p_\theta(x)$
  - Marginal likelihood estimator

$$\frac{1}{p_\theta(x_i)} = \int \frac{q(z)}{p_\theta(x_i)} dz = \frac{\int q(z) \cdot \frac{p_\theta(x_i,z)}{p_\theta(x_i,z)} dz}{p_\theta(x_i)} = \int \frac{p_\theta(x_i,z)}{p_\theta(x_i)} \cdot \frac{q(z)}{p_\theta(x_i,z)} dz$$

$$= \int p_\theta(z|x_i) \frac{q(z)}{p_\theta(x_i,z)} dz \approx \frac{1}{L} \sum_l \frac{q_\phi(z^{(l)})}{p_\theta(z) p_\theta(x_i|z^{(l)})}$$

  - In VAE, marginal likelihood estimator cannot be evaluated because it requires $l > 1$
  - If it was forced to evaluate the marginal likelihood in VAE, the result is almost same as loss function, it doesn't give any special meaning.

# Marginal Likelihood Estimator (2)

- **Gradient MCMC**
  - Max Welling and published "Bayesian learning via stochastic gradient langevin dynamics (2011), ICML2011"

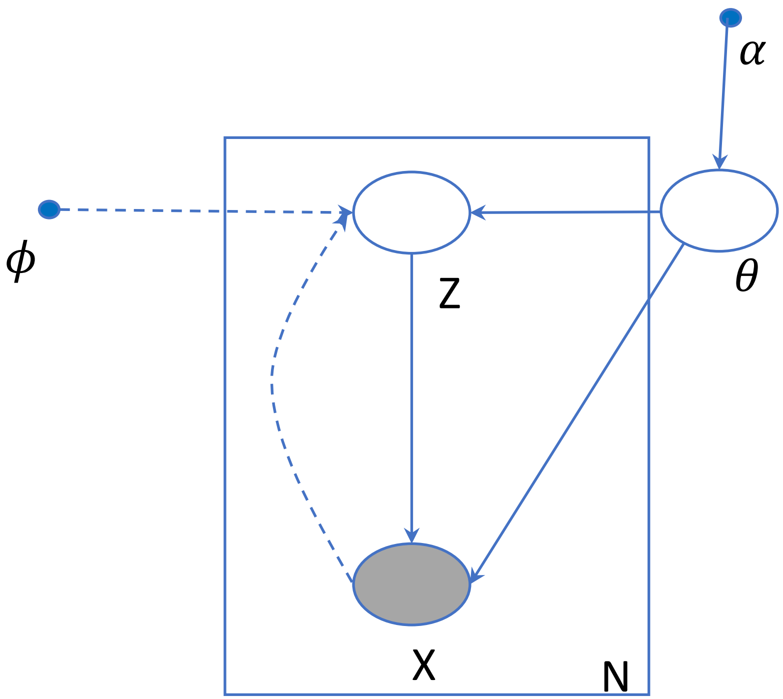
$$\Delta\theta_t = \frac{\epsilon_t}{2}(\nabla logp(\theta_t) + \frac{N}{n}\sum_{i=1}^{n}\nabla logp(x_i|\theta_t)) + \eta_t$$
$$where\ \eta_t \sim N(0, \epsilon_t)$$

Update (stochastic gradient descent)

$$\theta = \theta_t + \epsilon_t' \cdot \Delta\theta_t$$

  - In this paper, $\nabla \log p(z_t)$ , $\nabla \log p(x_i|z_t)$ can be evaluated.
  - Under this method, marginal likelihood estimator is established.

# FULL VARIATIONAL BAYES




[ AEVB algorithm ]

- **Framework**
  - $p_\theta(z) \sim N(0, I),\ where\ z \in R^J, I \in R^{J\times J}$
  - $p_\alpha(\theta) \sim N(0, I),\ where\ \theta \in R^M, I \in R^{M\times M}$
  - $p_\theta(x|z) \sim N\big(u_\theta(z), \sigma_\theta(z)\big)\ or\ Bern(\mathrm{p}_\theta(z))$
  - $q_\phi(z|x) \sim N(u_\phi(x), diag_\phi(x))$

# Full variational Bayes

- **Variational Bound**

$$ln p_\theta(X) \approx \int \left(lnP_\alpha(X|\theta) + lnP_\alpha(\theta) - lnq_\phi(\theta)\right) \cdot q_\theta(\theta)d\theta$$
$$= \frac{1}{L}\sum_{l=1}^{L} E[lnP_\alpha(X|\theta)] + \frac{1}{2}\sum_{M=1}^{M}\{1 + ln\sigma_{\theta\ m}^{2\ (l)} - \mu_{\theta\ m}^{2\ (l)} - \sigma_{\theta_m}^{2\ \ (l)}\ \}$$

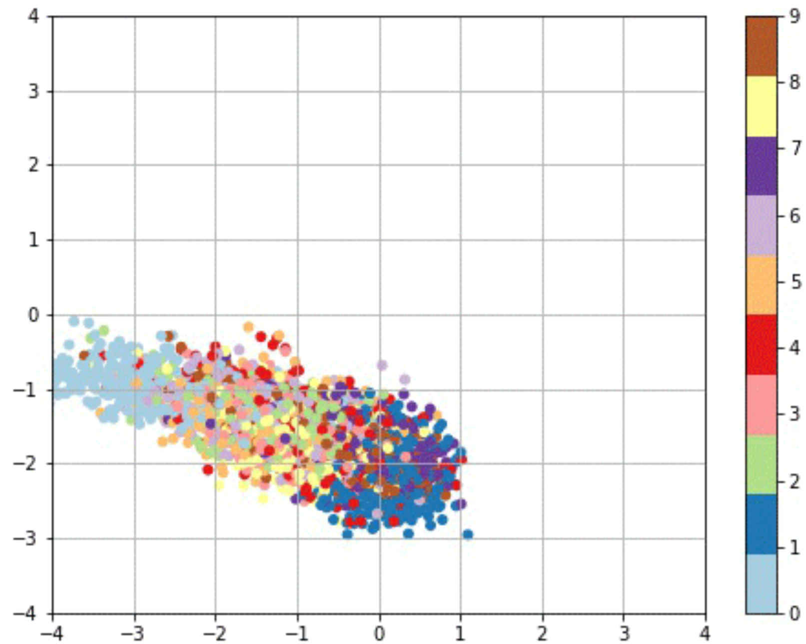
$$lnP_\alpha(X|\theta) = \int log\frac{p(x,z)}{q(z)} \cdot q(z)dz\ - \int log\frac{q(z)}{p(z|x)} \cdot q(z)dz$$
$$\approx \frac{1}{K}\sum_{i=1}^{N}(\sum_{k=1}^{K} \ln p_\theta(x_i|z_{i,j}) + \frac{1}{2}\sum_{j=1}^{J}\{1 + ln\sigma_{z\ i,j}^{2\ (k)} - \mu_{z\ i,j}^{2\ (k)} - \sigma_{z_{i,j}}^{2\ \ (k)}\ \})$$

- In this paper, $L = K = 1$ and $\ln p_\theta(x_i|z_{i,j}) = lnp_\theta(x|z), \forall x_i$

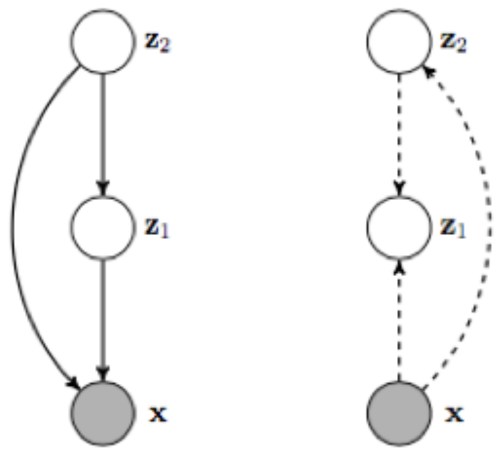
$$\therefore N \cdot lnp(x|z) + \frac{N}{2}\cdot\sum_{j=1}^{J}\{1 + ln\sigma_{z\ i,j}^{2\prime} - \mu_{z\ i,j}^{2\prime} - \sigma_{z_{i,j}}^{2\ \prime}\ \} + \frac{1}{2}\sum_{M=1}^{M}\{1 + ln\sigma_{\theta\ m}^{2\ (l)} - \mu_{\theta\ m}^{2\ (l)} - \sigma_{\theta_m}^{2\ \ (l)}\ \}$$

# EXPERIMENTAL RESULT

# Standard VAE



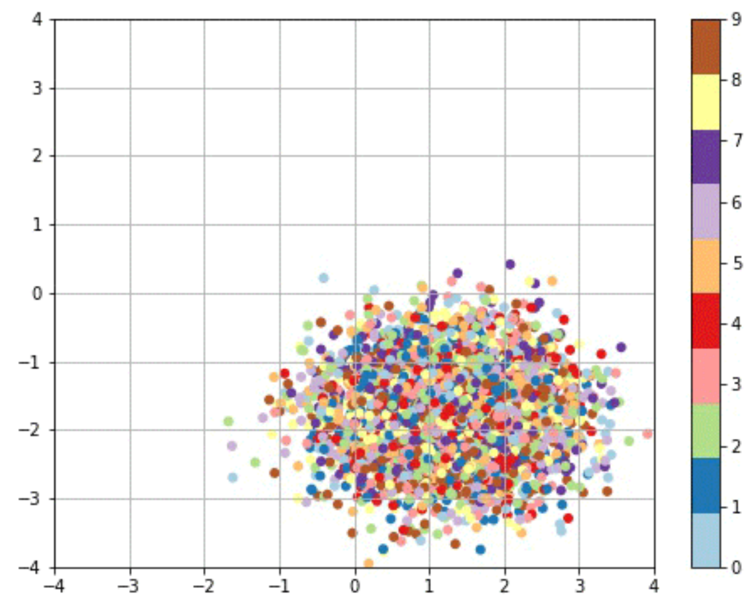
# Hierarchical VAE




generative part variational part

(b)

$$p_\lambda(\mathbf{z}_1|\mathbf{z}_2) = \mathcal{N}\big(\mathbf{z}_1|\mu_\lambda(\mathbf{z}_2), \mathrm{diag}(\sigma^2_\lambda(\mathbf{z}_2))\big),$$
$$q_\phi(\mathbf{z}_1|\mathbf{x}, \mathbf{z}_2) = \mathcal{N}\big(\mathbf{z}_1|\mu_\phi(\mathbf{x}, \mathbf{z}_2), \mathrm{diag}(\sigma^2_\phi(\mathbf{x}, \mathbf{z}_2))\big),$$
$$q_\psi(\mathbf{z}_2|\mathbf{x}) = \mathcal{N}\big(\mathbf{z}_2|\mu_\psi(\mathbf{x}), \mathrm{diag}(\sigma^2_\psi(\mathbf{x}))\big)$$



# Experimental Results

- **Table**

| Epoch | VAE | | Hierarchical VAE | |
|---|---|---|---|---|
| | Variational Bound | Log Marginal Likelihood | Variational Bound | Log Marginal Likelihood |
| 1 | 212.87 | -210.73 | 214.06 | -214.11 |
| 50 | 152.83 | -151.75 | 152.75 | -152.85 |
| 100 | 154.24 | -153.59 | 149.95 | -150.15 |
| 150 | 152.32 | -152.15 | 139.21 | -139.29 |
| 200 | 138.41 | -137.42 | 132.70 | -132.82 |

# WHAT'S NEXT?

CURRENT RESEARCH

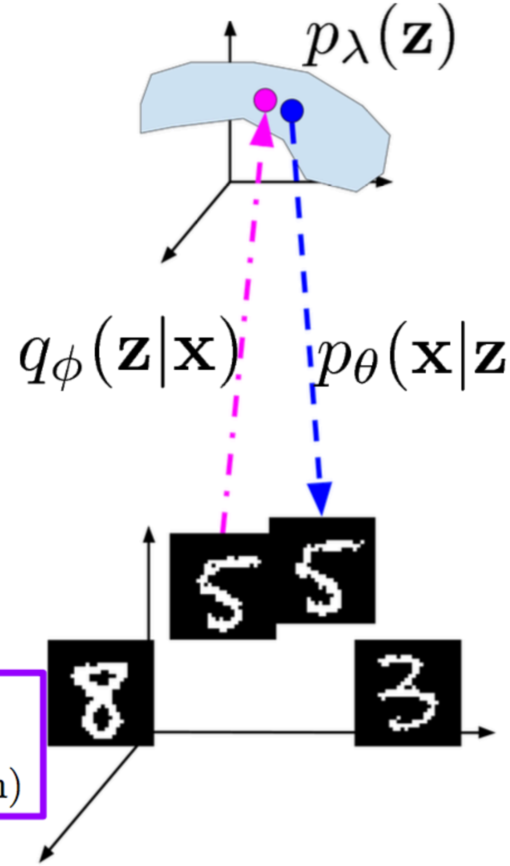
# DGM: Variational Auto-Encoder

$$q_\phi(\mathbf{z}|\mathbf{x}) \propto p_\theta(\mathbf{x}|\mathbf{z})\, p_\lambda(\mathbf{z})$$

Normalizing flows
Volume-preserving flows
Gaussian processes
Stein Particle Descent
Operator VI

Feedforward nets
Convolutional nets
PixelCNN
Gated PixelCNN

Auto-regressive Prior (Chen et al.)
Objective Prior (Nalisnick & Smyth)
Stick-Breaking Prior (Nalisnick & Smyth)



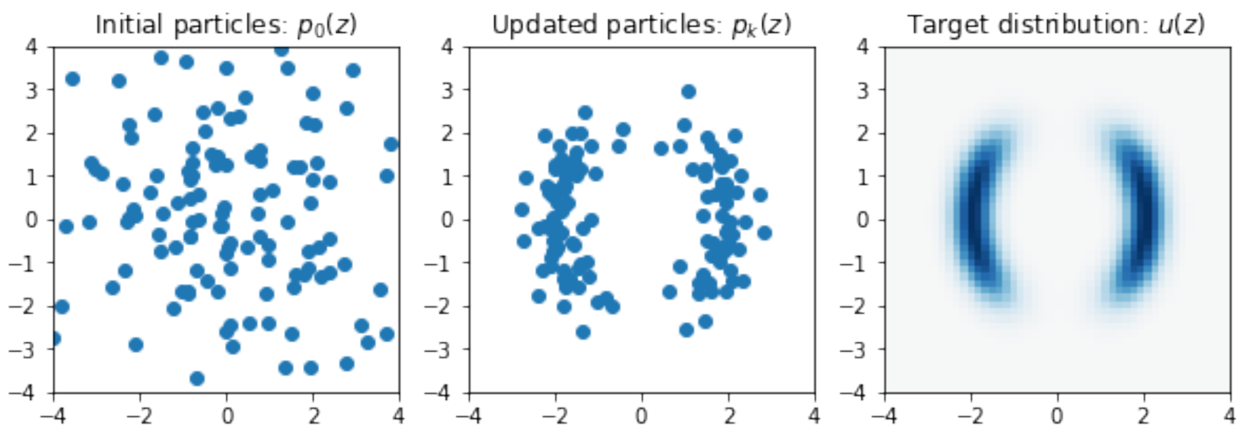

J. Tomczak (2018) VAM with a Vamp Prior at MPI Tubingen presentation

# Normalizing Flow / Stein Method

- **Iterative update : $z_i \sim q_\phi(z_i|x_i)$**
  - Based on change of variable
    - Objective : minimizing $KLD(q||p)$
    - Normalizing flow : using tractable determinant of Jacobian
    - Stein method : using the RKHS kernel to describe direction for minimizing KLD

$$\boldsymbol{T}_l^*(x) = x + \epsilon_l \boldsymbol{\phi}_{q_l,p}^*(x).$$

$$q_0 \xrightarrow{\boldsymbol{T}_0^*} q_1 \xrightarrow{\boldsymbol{T}_1^*} q_2 \xrightarrow{\boldsymbol{T}_2^*} \cdots \to q_\infty = p(\boldsymbol{\phi}_{q_l,p}^* = 0)$$



# VAE WITH VAMPPRIOR

- **Main Idea**

  -. Prior Distribution = Aggregate Posterior Distribution

$$P_\lambda(z) = \frac{1}{K}\sum_{k=1}^{K} q(z|x)$$

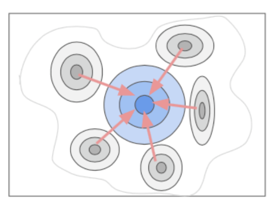
  -. Pseudo inputs

  . Users should select learnable pseudo inputs.

  . If pseudo inputs are randomly chosen, we couldn't expect better performance.

  -. Expectation for this approach

$$\text{min. } \mathbb{H}[q(\mathbf{z})] + KL[q(\mathbf{z})||p_\lambda(\mathbf{z})]$$



Standard prior is too strong and overregularizes the encoder.

What is the "optimal" prior?

# VAE WITH VAMPPRIOR

- **Variational Bound**

$$\mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\ln p(\mathbf{x})\big] \geq \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\Big[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z})]\Big] + \\ + \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\Big[\mathbb{H}[q_\phi(\mathbf{z}|\mathbf{x})]\Big] + \\ - \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})]$$

$$\max_{p_\lambda(\mathbf{z})} -\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})] + \beta\Big(\int p_\lambda(\mathbf{z})\mathrm{d}\mathbf{z} - 1\Big)$$

$$\boxed{p_\lambda^*(\mathbf{z}) = \frac{1}{N}\sum_{n=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}_n)}$$

# VAE WITH VAMPPRIOR

- **Experimental Result**

| DATASET | VAE ($L = 1$) standard | VAE ($L = 1$) VampPrior | HVAE ($L = 2$) standard | HVAE ($L = 2$) VampPrior | CONVHVAE ($L = 2$) standard | CONVHVAE ($L = 2$) VampPrior | PIXELHVAE ($L = 2$) standard | PIXELHVAE ($L = 2$) VampPrior |
|---|---|---|---|---|---|---|---|---|
| staticMNIST | −88.56 | **−85.57** | −86.05 | **−83.19** | −82.41 | **−81.09** | −80.58 | **−79.78** |
| dynamicMNIST | −84.50 | **−82.38** | −82.42 | **−81.24** | −80.40 | **−79.75** | −79.70 | **−78.45** |
| Omniglot | −108.50 | **−104.75** | −103.52 | **−101.18** | −97.65 | **−97.56** | −90.11 | **−89.76** |
| Caltech 101 | −123.43 | **−114.55** | −112.08 | **−108.28** | −106.35 | **−104.22** | **−85.51** | −86.22 |
| Frey Faces | 4.63 | **4.57** | 4.61 | **4.51** | 4.49 | **4.45** | 4.43 | **4.38** |
| Histopathology | 6.07 | **6.04** | 5.82 | **5.75** | 5.59 | **5.58** | 4.84 | **4.82** |

Table 2: Test LL for static MNIST.

| MODEL | LL |
|---|---|
| VAE ($L = 1$) + NF [32] | −85.10 |
| VAE ($L = 2$) [6] | −87.86 |
| IWAE ($L = 2$) [6] | −85.32 |
| HVAE ($L = 2$) + SG | −85.89 |
| HVAE ($L = 2$) + MoG | −85.07 |
| HVAE ($L = 2$) + VAMPPRIOR *data* | −85.71 |
| HVAE ($L = 2$) + VAMPPRIOR | **−83.19** |
| AVB + AC ($L = 1$) [28] | −80.20 |
| VLAE [7] | **−79.03** |
| VAE + IAF [18] | −79.88 |
| CONVHVAE ($L = 2$) + VAMPPRIOR | −81.09 |
| PIXELHVAE ($L = 2$) + VAMPPRIOR | −79.78 |

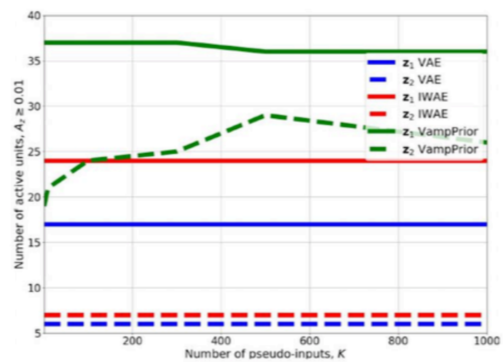



Figure 3: A comparison between two-level VAE and IWAE with the standard normal prior and theirs VampPrior counterpart in terms of number of active units for varying number of pseudo-inputs on static MNIST.

# Any Questions?

# References


- Kingma and Welling(2013), *Auto-encoding Variational Bayes*, ICLR2014
- Bishop(2006), Pattern recognition and Machine learning
- Rezende, Mohamed and Wiestra(2014) *Stochastic Backpropagation and Approximate Inference in Deep Generative Models*, ICML2014
- Liu and Wang(2016), *Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm,* NIPS2016
- Tomczak and Welling(2018), *VAE with a VampPrior*, AISTAT2018

J. Tomczak (2018) VAM with a Vamp Prior at MPI Tubingen presentation