Auto-Encoding Variational Bayes

Critical Summary

Natnael Daba

March 13, 2024

Authors

Dr. Diederik P. Kingma
Research Scientist
Google Brain

Prof. Max Welling
Distinguished Scientist
Microsoft Research

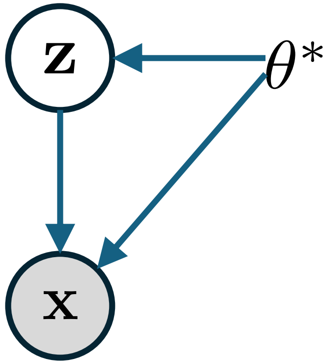
Content

Assumptions

Given dataset $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ of N i.i.d data samples $\mathbf{x}$

We assume the following:

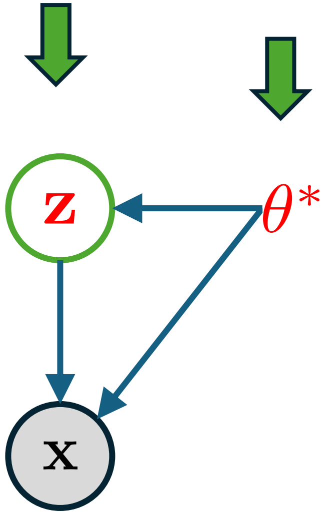
- Some random process generated the data.
- This process involves an unobserved cont. r.v $\mathbf{z}$
- The process is as follows:
  1. $\mathbf{z}^{(i)}$ is generated from prior $p_{\boldsymbol{\theta}^*}(\mathbf{z})$
  2. $\mathbf{x}^{(i)}$ is generated from conditional $p_{\boldsymbol{\theta}^*}(\mathbf{x}|\mathbf{z})$
- Prior and likelihood come from parametric family of distributions $p_{\boldsymbol{\theta}}(\mathbf{z})$ and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$



$\boldsymbol{\theta}^*$: true parameters
$\mathbf{z}^{(i)}$: latent variable

But first …

We have a problem!



We don't know the complete data generation process.

- $\boldsymbol{\theta}^*$ and $\mathbf{z}^{(i)}$ are unknown to us

Three problems:

1. Efficient approximate ML or MAP estimation for $\boldsymbol{\theta}$
2. Efficient approximate posterior inference of $\mathbf{z}$ given $\mathbf{x}$
3. Efficient approximate marginal inference of $\mathbf{x}$

Notation and terminology

$q_\phi(\mathbf{z}|\mathbf{x})$: recognition model or probabilistic encoder

- Approximation to the intractable posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$

ϕ : variational parameters

$\boldsymbol{\theta}$: generative model parameters

$\mathbf{z}$: latent representation or *code*

$p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$: probabilistic decoder

The variational lower bound

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}) = \textstyle\sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$$

(log marginal likelihood)

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)},\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)})}\right] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)},\mathbf{z})q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)})}\right] \\
&= \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\right]}_{\substack{= \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \\ \text{(ELBO)}}} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}{p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)})}\right]}_{= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)}))}
\end{aligned}$$

The variational lower bound

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - \underbrace{D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)}))}_{\geq 0}$$

KL divergence is non-negative

$$\leq \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$$

$D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)}))$ determines two distances:

1. KL divergence of the approximate posterior from the true posterior;

2. Gap between the ELBO $\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})$ and the marginal likelihood $\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$; also called the *tightness* of the bound.

The variational lower bound

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\right]$$

$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z})\right]$$

$$= -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}))\right]$$

Goal: differentiate and optimize the lower bound $\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})$ w.r.t. both ϕ and $\boldsymbol{\theta}$

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \mathbb{E}_{q_\phi(\mathbf{z})}[f(\mathbf{z})\nabla_\phi \log q_\phi(\mathbf{z})] \simeq \underbrace{\frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z}^{(l)})\nabla_\phi \log q_\phi(\mathbf{z}^{(l)})}_{\text{Naïve Monte Carlo gradient estimator}}$$

where $\mathbf{z}^{(l)} \sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$

exhibits very high variance

Stochastic Gradient Variational Bayes (SGVB) estimator

TL;DR: SGVB is an unbiased estimator of the lower bound without the high variance issue.

Reparametrize the r.v. $\tilde{\mathbf{z}} \sim q_\phi(\mathbf{z}|\mathbf{x})$ as:

$$\tilde{\mathbf{z}} = \underbrace{\boxed{g_\phi(\boldsymbol{\epsilon}, \mathbf{x})}}_{\text{Differentiable transformation}} \text{ with } \underbrace{\boxed{\boldsymbol{\epsilon}}}_{\text{Noise variable}} \sim p(\boldsymbol{\epsilon})$$

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[f(\mathbf{z})\right] = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[f(g_\phi(\boldsymbol{\epsilon}, \mathbf{x}^{(i)}))\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(g_\phi(\boldsymbol{\epsilon}^{(l)}, \mathbf{x}^{(i)}))$$

where $\boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$

SGVB estimator

Remember our ELBO:

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z})\right]$$

Applying reparameterization of $\mathbf{z}$ to our ELBO, we get our SGVB estimator $\tilde{\mathcal{L}}^A(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) \simeq \mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})$:

$$\tilde{\mathcal{L}}^A(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = \frac{1}{L}\sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z}^{(i,l)}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}^{(i,l)}|\mathbf{x}^{(i)})$$

where $\mathbf{z}^{(i,l)} = g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}^{(i,l)}, \mathbf{x}^{(i)})$ and $\boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$

SGVB estimator

Remember our *second* ELBO variant:

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}))\right]$$

Applying reparameterization of $\mathbf{z}$ to our *second* ELBO, we get:

$$\tilde{\mathcal{L}}^B(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \frac{1}{L}\sum_{l=1}^{L}(\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}))$$

where $\mathbf{z}^{(i,l)} = g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}^{(i,l)}, \mathbf{x}^{(i)})$ and $\boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$

which is our second version of the SGVB estimator

$$\tilde{\mathcal{L}}^B(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) \simeq \mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})$$

SGVB estimator (full dataset)

Given dataset $\mathbf{X}$ with N datapoints:

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{X}) \simeq \tilde{\mathcal{L}}^M(\boldsymbol{\theta}, \phi; \boxed{\mathbf{X}^M}) = \boxed{\frac{N}{M} \sum_{i=1}^{M} \tilde{\mathcal{L}}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)})}$$

$\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{M}$

Minibatch of M data samples

Estimator of ELBO of the full dataset

We can now take derivatives $\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \tilde{\mathcal{L}}(\boldsymbol{\theta}, \phi; \mathbf{X}^M)$ and optimize ϕ and $\boldsymbol{\theta}$

The AEVB algorithm

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 $\boldsymbol{\theta}, \boldsymbol{\phi}$

The reparameterization trick

Our second ELBO variant:

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \boxed{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}))\right]}$$

Involves sampling $\mathbf{z}$ from $q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$ i.e. $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$

But sampling is a stochastic process and therefore we cannot backpropagate gradients through it.

Solution: express $\mathbf{z}$ as a deterministic variable $\mathbf{z} = g_\phi(\boldsymbol{\epsilon}, \mathbf{x})$

where $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$ and $g_\phi(.)$ is some vector-valued function parametrized by ϕ

The reparameterization trick

Example: let $q_\phi(\mathbf{z}|\mathbf{x}^{(i)})$ be a multivariate Gaussian with diagonal covariance structure:

$$\mathbf{z} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{2(i)}\mathbf{I})$$

$$\mathbf{z}^{(i)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$; Reparameterization trick.

where $\odot$ denotes element-wise product.

The reparameterization trick

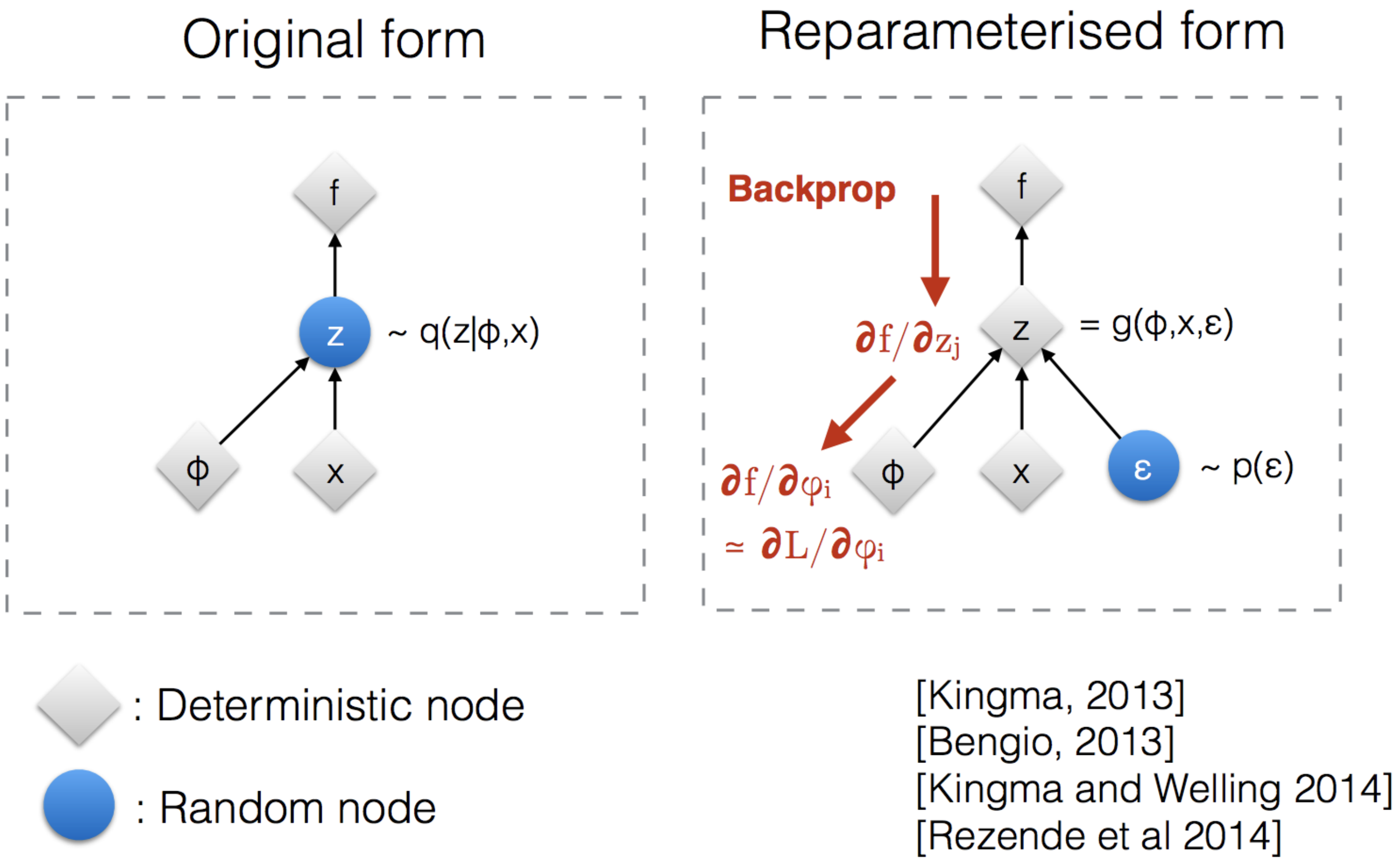


Illustration of how the reparameterization trick makes the sampling process trainable.(Image source: Slide 12 in Kingma's NIPS 2015 workshop talk)

Example: Variational Auto-Encoder

Idea: use a neural network for $q_\phi(\mathbf{z}|\mathbf{x})$ and optimize ϕ and $\boldsymbol{\theta}$ jointly using the AEVB algorithm.

Let:

$$p_{\boldsymbol{\theta}}(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$$

$p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$: multivariate Gaussian or Bernoulli

- Parameters of this distribution are computed from $\mathbf{z}$ with a MLP

$p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$: true (but intractable) posterior

- Assume this takes on approx. Guassian with an approx. diagonal covariance.

Example: Variational Auto-Encoder

$q_\phi(\mathbf{z}|\mathbf{x})$: variational approximate posterior

- We can let this be a multivariate Gaussian with a diagonal covariance structure.

$$\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)}) = \log \mathcal{N}(\mathbf{z}; \boxed{\boldsymbol{\mu}^{(i)}}, \boxed{\boldsymbol{\sigma}^{2(i)}}\mathbf{I})$$

Outputs of encoding MLP

Note: in this model $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_{\boldsymbol{\theta}}(\mathbf{z})$ are Gaussian and thus $D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}))$ has a closed form

Example: Variational Auto-Encoder

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) \simeq \frac{1}{2} \sum_{j=1}^{J} \left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + \frac{1}{L} \sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)} | \mathbf{z}^{(i,l)})$$

where

$$\mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)} \text{ and } \boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Example: Variational Auto-Encoder

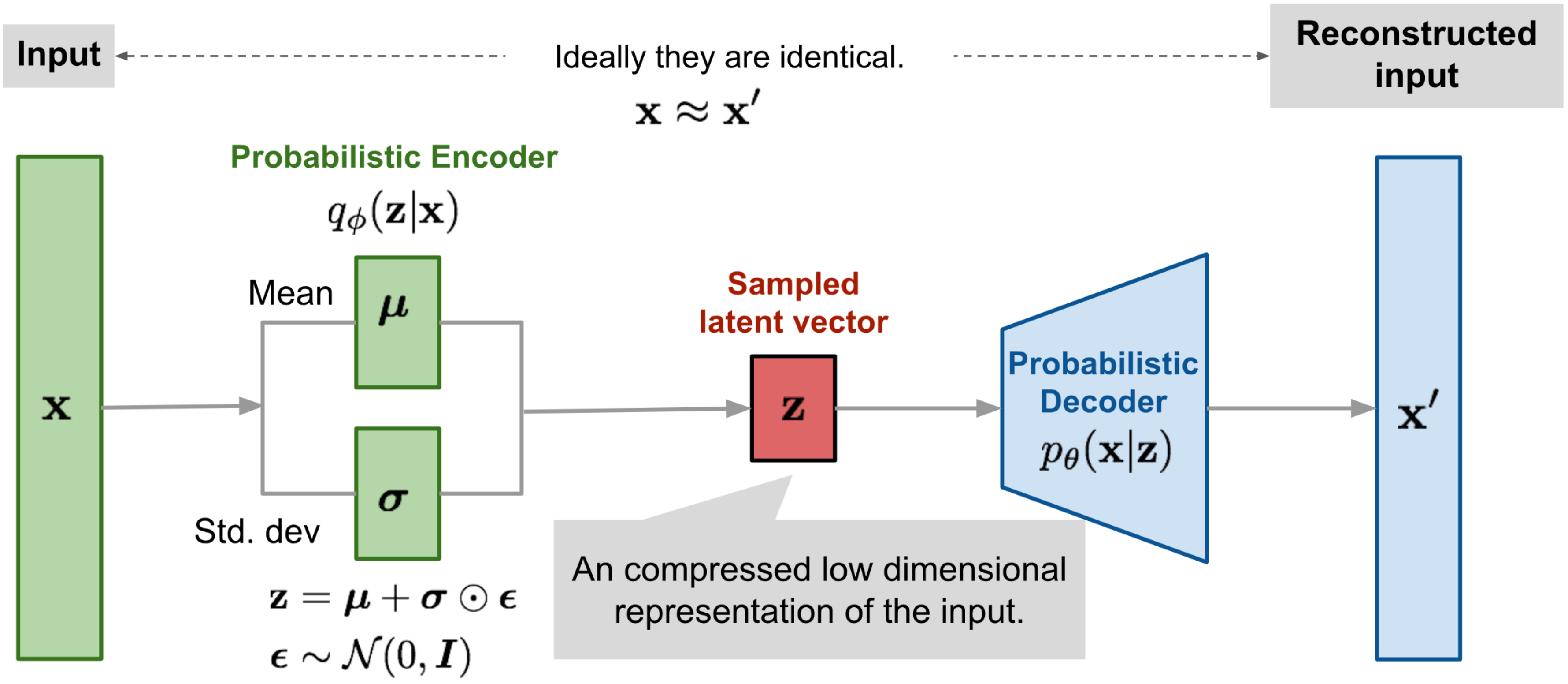
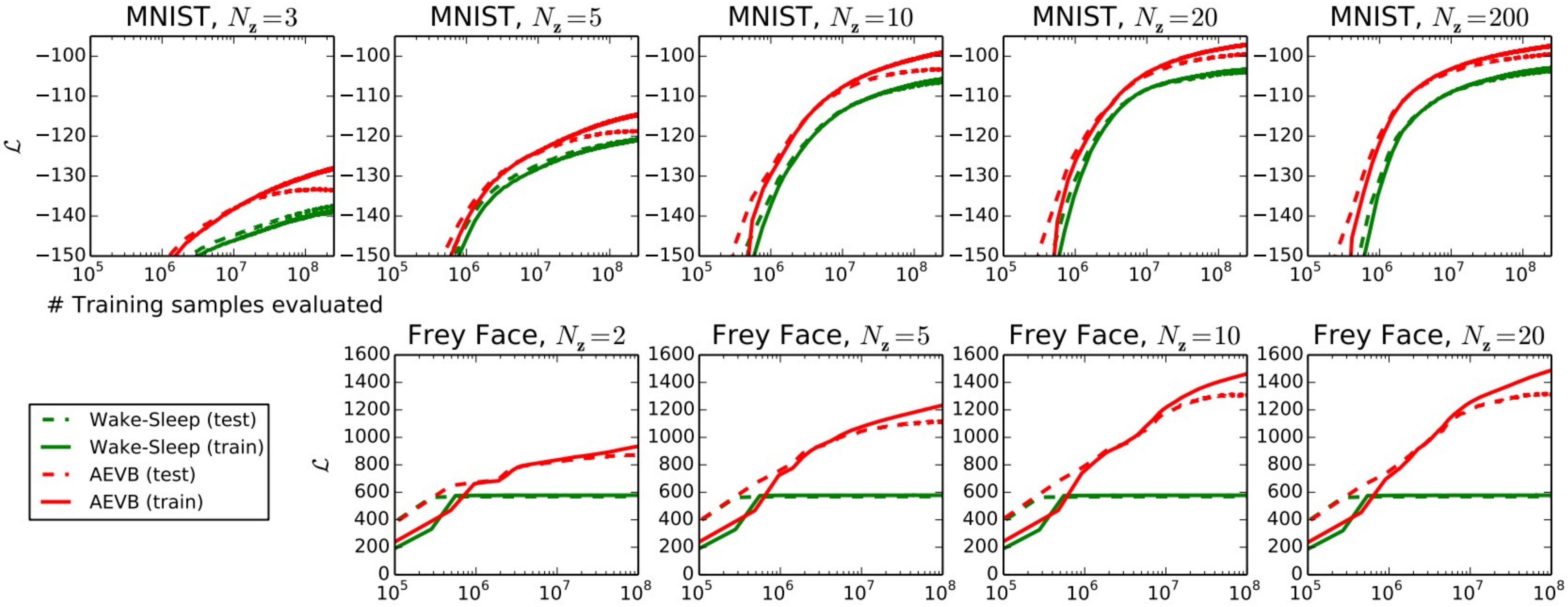


Illustration of variational autoencoder model with the multivariate Gaussian assumption.(source: https://lilianweng.github.io/posts/2018-08-12-vae/)

Experiments

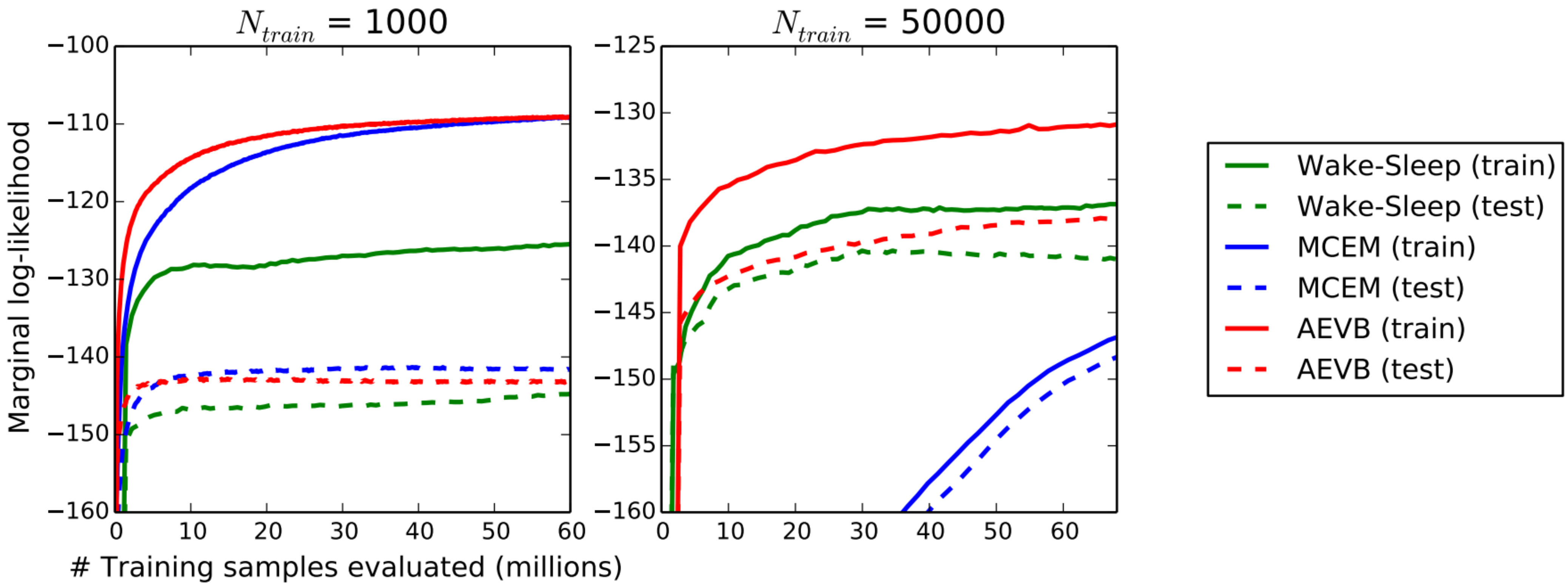
Tasks:
1. Train generative models of images from MNIST and Frey Face datasets
2. Compare learning algorithms in terms of:
   a) The variational lower bound
   b) The estimated marginal likelihood

Results: likelihood lower bound

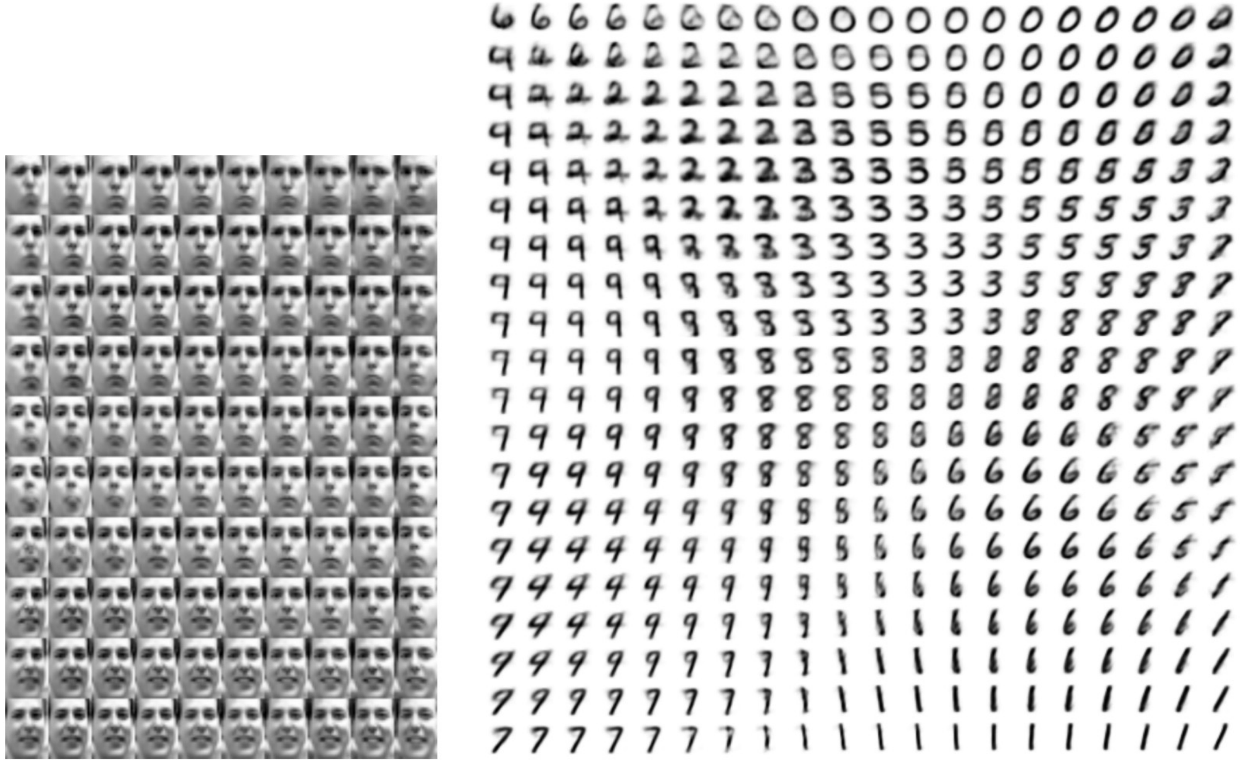


Vertical axis: the estimated average variational lower bound per datapoint. The estimator variance was small (< 1) and omitted. **Horizontal axis**: amount of training points evaluated. N_z : dim. of latent space

Results: marginal likelihood



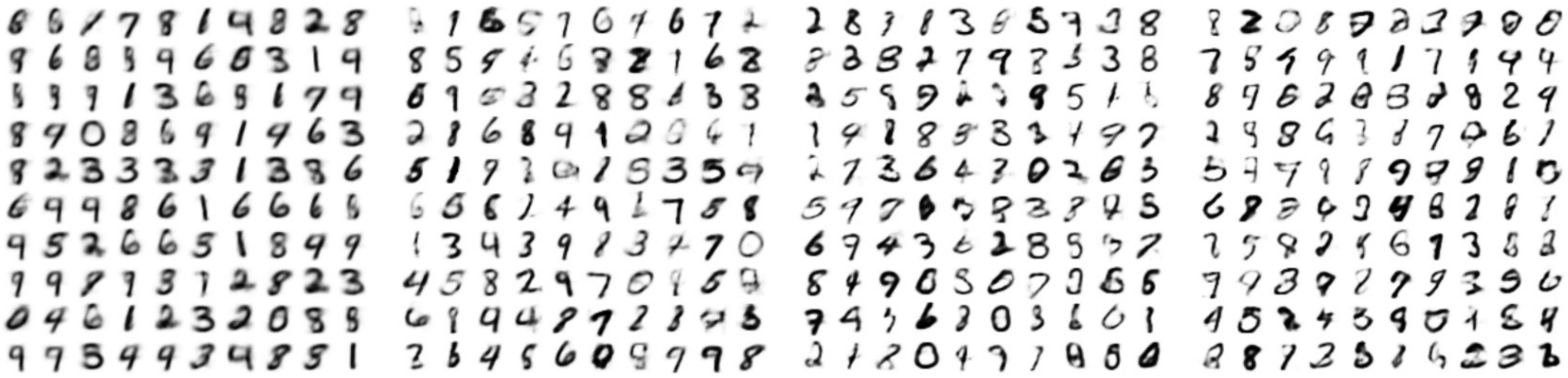
Results: Visualization of high-dimensional data



(a) Learned Frey Face manifold (b) Learned MNIST manifold

Figure 4: Visualisations of learned data manifold for generative models with two-dimensional latent space, learned with AEVB. Since the prior of the latent space is Gaussian, linearly spaced coordinates on the unit square were transformed through the inverse CDF of the Gaussian to produce values of the latent variables $\mathbf{z}$. For each of these values $\mathbf{z}$, we plotted the corresponding generative $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ with the learned parameters $\boldsymbol{\theta}$.

Results: Visualization of high-dimensional data



(a) 2-D latent space (b) 5-D latent space (c) 10-D latent space (d) 20-D latent space

Figure 5: Random samples from learned generative models of MNIST for different dimensionalities of latent space.

Code

- https://github.com/AntixK/PyTorch-VAE/blob/master/models/vanilla_vae.py

Thank you