ENGR-E 533 “Deep Learning Systems”
Lecture 12: Generative Models

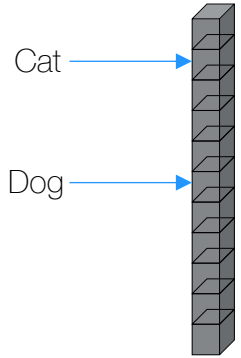
Minje Kim
Department of Intelligent Systems Engineering
Email: minje@indiana.edu
Website: http://minjekim.com
Research Group: http://saige.sice.indiana.edu
Meeting Request: http://doodle.com/minje

INDIANA UNIVERSITY
SCHOOL OF INFORMATICS,
COMPUTING, AND ENGINEERING

Latent Representation of Data

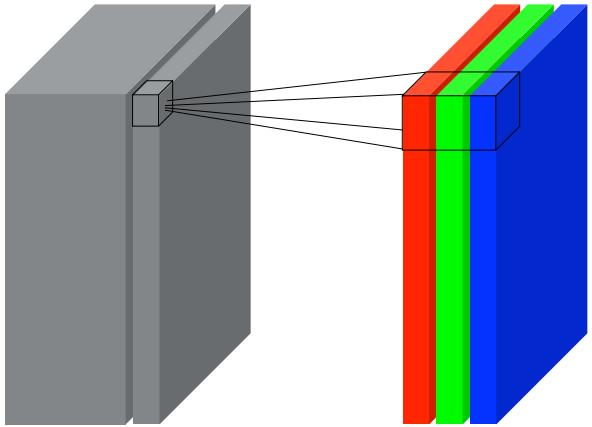
- A decoder made of deconvolution layers

- Suppose I want to create an image of a cat
  - What should the code vector look like?
    - Perhaps a one-hot vector?
- How about a dog?
- Any problem with this?



• • •

Deconvolution Layers



A colorful cat image

  - We can't really distinguish different cants
  - We can't really differentiate cats and lions versus cats and dogs

Latent Representation of Data

- Latent embedding vector

- How about this?

0	0.81	0	0	0	0	0	0	0	0	← A cat
0	0.78	0	0	0	0	0	0	0	0	← Another cat

- An even better representation

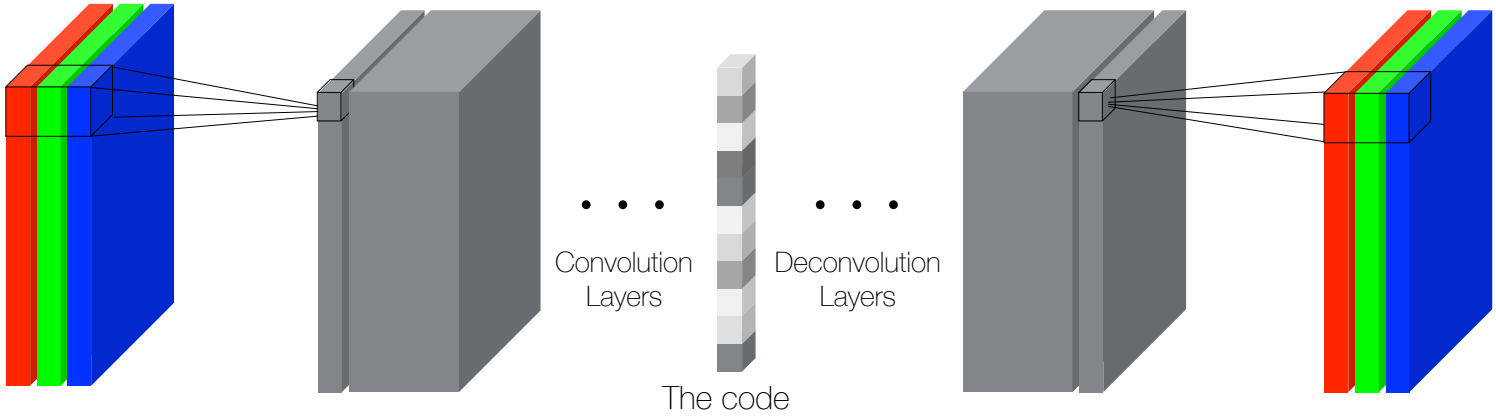
0	0.81	0	0	0	0.1	0	0	0	0	← A cat a little looking like a dog
0	0.78	0	0	0	0.55	0	0	0	0	← Another cat looking a lot like a dog

  - And so on

- Eventually we need a latent representation with real numbers

Autoencoders with A Priori Knowledge

- Regularized autoencoders



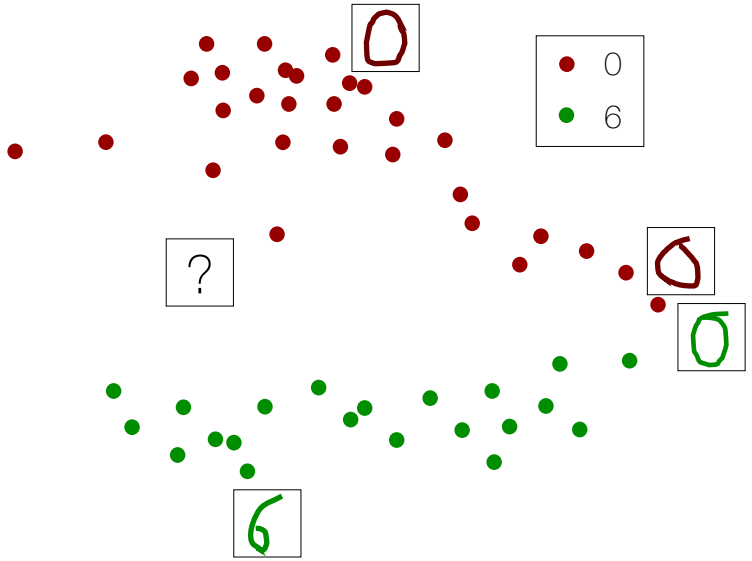
- What do we want from the codes?
  - Minimal loss of information
- How do you define “information”?
  - The decoder should generate a similar output to the input
  - Originally similar input examples share similar codes
  - **As a generative model**
    - **Should be able to create examples from unseen codes**

Autoencoders with A Priori Knowledge

- Regularized autoencoders

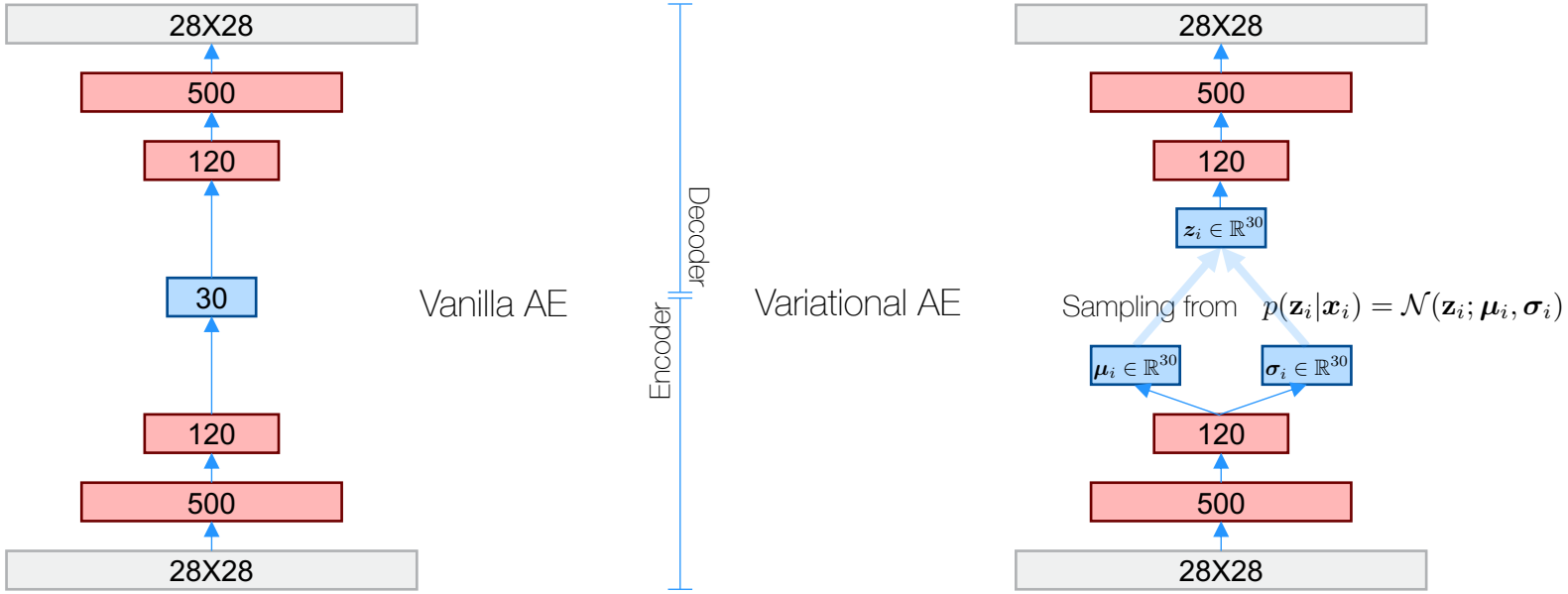
- Problems with vanilla AEs
  - No control over the latent variables
  - Why do we need the control in the first place?
- A sample in the latent space
  - e.g. Could be something in between the two classes
  - Can we be more specific?
- The p.d.f. of the latent variables given the data set

  $p(\mathbf{z}|\boldsymbol{X})$
  - We want to know it better
    - This could be a nasty one
  - Furthermore, we want it to be more intuitively distributed
    - e.g. z_1 is for thickness; z_2 is for rotation; etc.
    - Why?
  - How could you "generate" a new example with a choice of thickness if the p.d.f. is too complicated?



Variational AutoEncoders

- Latent variables are from multivariate Normal distributions

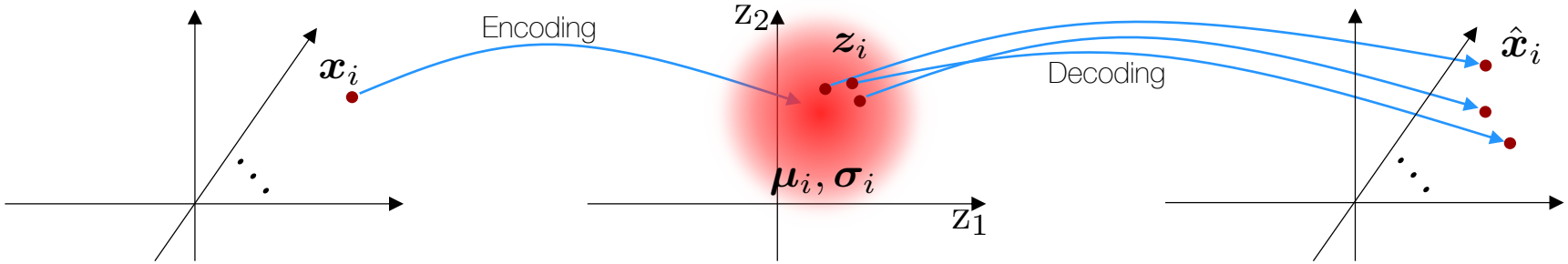


- Why this weird structure?
  - You want an easy distribution for $p(\mathbf{z}_i|\boldsymbol{x}_i)$
  - But it makes training difficult—needs a special structure

Variational AutoEncoders

- Latent variables are from multivariate Normal distributions

Sampling turns decoding into a stochastic process,
enabling it to learn from multiple samples of the LV



Parameterization of the Normal distribution
lets encoding result in a distribution over LV

- During training
  - Every epoch the reconstructed input (output of the decoder) is from a sampled version of the latent variable
  - The training algorithm is exposed to the continuous distribution of the LV, which is not possible with vanilla AE
- During testing
  - The decoder itself is deterministic, but produces stochastic predictions due to the sampling process

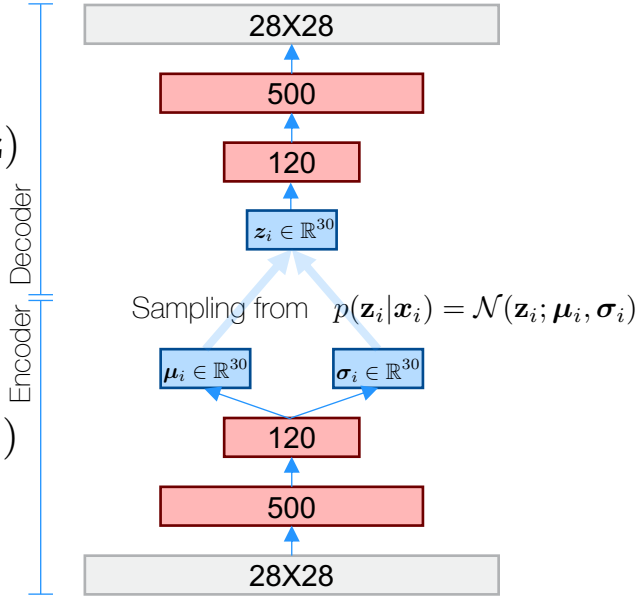
Variational AutoEncoders

- A plain explanation

- Let's call the encoder $q_\theta(\mathbf{z}|\boldsymbol{x})$
  - In English, it's the posterior distribution over the LV given the data sample
  - It's parameterized by the weights and biases of the network
  - It looks weird because originally we wanted $p(\mathbf{z}|\boldsymbol{x})$ but it's too hard to compute (we'll revisit this issue)
- Let's call the decoder $p_\phi(\boldsymbol{x}|\mathbf{z})$
  - Distribution over the multivariate observations (pixels) given the LV
- They are all neural networks, anyway
- The loss function for an example

$$\mathcal{L}_i(\theta,\phi) = \underbrace{-\mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\boldsymbol{x}_i)}}_{\text{Expectation over the LV}}\underbrace{[\log p_\phi(\boldsymbol{x}_i|\mathbf{z})]}_{\text{Likelihood of observing the input given the LV (decoder gives you the distribution)}} + \underbrace{KL(q_\theta(\mathbf{z}|\boldsymbol{x}_i)||p(\mathbf{z}))}_{\text{Regularlization: the LV should be close to its prior, usually } \mathcal{N}(0,1)}$$

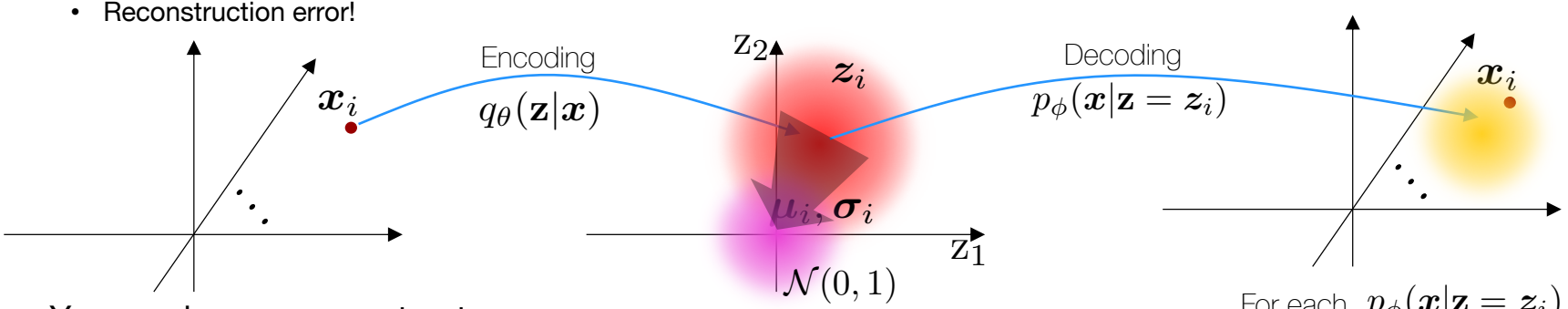
  - So, this is an AE with regularization



Variational AutoEncoders

- A plain explanation

- A more precise animation based on the loss
- Each sample z_i deterministically defines a distribution over the original data
  - Collect this log likelihoods and get their expected value with $q_\theta(\mathbf{z}|\boldsymbol{x})$ via sampling
    - Reconstruction error!



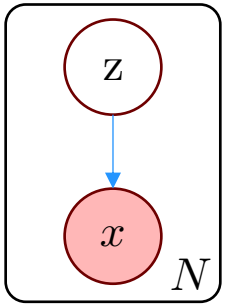
For each $p_\phi(\boldsymbol{x}|\mathbf{z}=\boldsymbol{z}_i)$ you can calculate the likelihood of the input $\boldsymbol{x}_i$

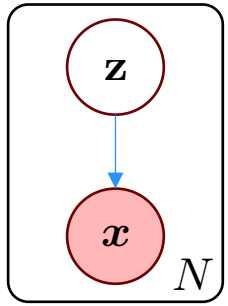
- You need some more structure
  - You want $q_\theta(\mathbf{z}|\boldsymbol{x})$ to look like the standard normal distribution. Why?
  - In VAE, each input example defines a Gaussian distribution
    - In Vanilla AE, they can freely define whatever latent distribution that minimizes the reconstruction error
    - Recall we wanted some control over the LV so that we can synthesize new examples

Variational AutoEncoders

- Probabilistic understanding

- Data generation process from a latent variable
  - Draw a sample from the LV $z_i \sim p(\mathrm{z})$
  - Then, draw a sample from the conditional $x_i \sim p(\mathrm{x}|\mathrm{z} = z_i)$
- Multivariate case
  - $\boldsymbol{z}_i \sim p(\mathbf{z})$
  - $\boldsymbol{x}_i \sim p(\mathbf{x}|\mathbf{z} = \boldsymbol{z}_i)$
- Inference





Likelihood Prior

Posterior $p(\mathbf{z}|\mathbf{x}) = \dfrac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x})}$ ← Evidence

  - Evidence makes your life difficult

$$= \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{\int_{\mathbf{z}} p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}$$

← This integral can be very complicated

Variational AutoEncoders

- Probabilistic understanding

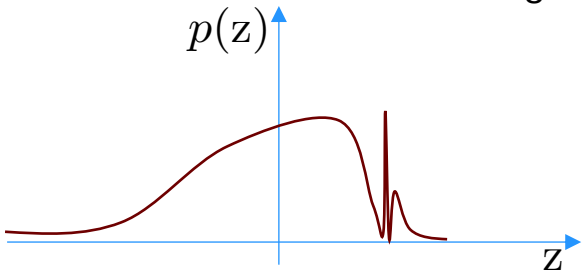
- Wait, wasn't it okay to ignore?
  - In Maximum A Posteriori (MAP) estimation

$$p(\mathbf{z}|\mathbf{x}) \propto p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$$

$$\arg\max_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}) = \arg\max_{\mathbf{z}} p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$$

  - e.g. You want to know the class label of your data sample that maximizes the posterior probability

- Finding the mode of the distribution is different from knowing the distribution



  - In VAE the encoder wants to estimate the posterior distribution properly, not just its mode

Variational Inference

- Evidence Lower BOund (ELBO)

- Variational Inference assumes an approximation to the posterior $q^*(\mathbf{z}) = \arg\min_{q(\mathbf{z})\in\mathcal{Q}} KL\big(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\big)$
  - So that the inference can be done with a more tractable distribution
- But this optimization is difficult, because

$$\begin{aligned} KL\big(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\big) &= \int_{q(\mathbf{z})} q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z}|\mathbf{x})} \\ &= \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] - \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{z}|\mathbf{x})] \\ &= \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] - \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x},\mathbf{z}) - \log p(\mathbf{x})] \\ &= \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] - \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x},\mathbf{z})] + \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x})] \end{aligned}$$

Need to calculate the evidence thing once again

- Instead, you can use Evidence Lower BOund (ELBO)

$$\begin{aligned} ELBO(q) &= -\mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] + \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x},\mathbf{z})] \\ &= -\mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] + \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] + \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{z})] \\ &= \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] - KL\big(q(\mathbf{z})||p(\mathbf{z})\big) \end{aligned}$$

  - Because $\log p(\mathbf{x})$ is a constant, **maximizing ELBO is equivalent to minimizing the KL divergence**
  - Or, you can **minimize the negative ELBO**: $-ELBO(q) = -\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] + KL\big(q(\mathbf{z})||p(\mathbf{z})\big)$
  - Ring a bell? The loss of VAE: $\mathcal{L}_i(\theta,\phi) = -\mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\boldsymbol{x}_i)}[\log p_\phi(\boldsymbol{x}_i|\mathbf{z})] + KL(q_\theta(\mathbf{z}|\boldsymbol{x}_i)||p(\mathbf{z}))$

Variational Inference

- Evidence Lower BOund (ELBO)

- This is a usual combination in Bayesian learning: maximize the likelihood along with a prior

$$-ELBO(q) = -\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] + KL\big(q(\mathbf{z})||p(\mathbf{z})\big)$$

- Another interpretation

$$\begin{aligned} KL\big(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\big) &= \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] - \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x},\mathbf{z})] + \log p(\mathbf{x}) \\ &= -ELBO + \text{evidence} \\ \leftrightarrow \text{evidence} &= \underbrace{KL\big(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\big)}_{\geq 0} + ELBO \end{aligned}$$

  - ELBO is the lowest possible value of the evidence, when the KL divergence is zero
  - It is achieved when $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x})$

- Relationship to EM
  - In EM we have parameters to estimate, which doesn't exist here in VI (i.e. evidence is conditioned on the parameters)
  - E-step minimizes $KL\big(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\big)$ by setting $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x})$
  - M-step maximizes ELBO w.r.t. the parameters

Variational Inference for VAE

- Mean-field approximation

- Mean-field approximation for VAE: another assumption that makes your life easier

$$q(\mathbf{z}) = \prod_i^N q(\mathbf{z}_i; \boldsymbol{\lambda}_i) = \prod_i^N q(\mathbf{z}_i; \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$$

  - Each data point has an associated LV, and its parameters are not shared with the other LVs

- ELBO with mean-field approximation

$$-ELBO(q) = -\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}|\mathbf{z})] + KL\big(q(\mathbf{z})||p(\mathbf{z})\big)$$

$$-ELBO(\boldsymbol{\lambda}_i) = -\mathbb{E}_{q_{\boldsymbol{\lambda}_i}(\mathbf{z}_i|\mathbf{x}_i)}[\log p(\mathbf{x}_i|\mathbf{z}_i)] + KL\big(q_{\boldsymbol{\lambda}_i}(\mathbf{z}_i|\mathbf{x}_i)||p(\mathbf{z})\big)$$

Mean-field factorization

- Neural net?

$$-ELBO_i(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\mathbb{E}_{q_{\boldsymbol{\theta}}(\mathbf{z}_i|\mathbf{x}_i)}[\log p_{\boldsymbol{\phi}}(\mathbf{x}_i|\mathbf{z}_i)] + KL\big(q_{\boldsymbol{\theta}}(\mathbf{z}_i|\mathbf{x}_i)||p(\mathbf{z})\big)$$

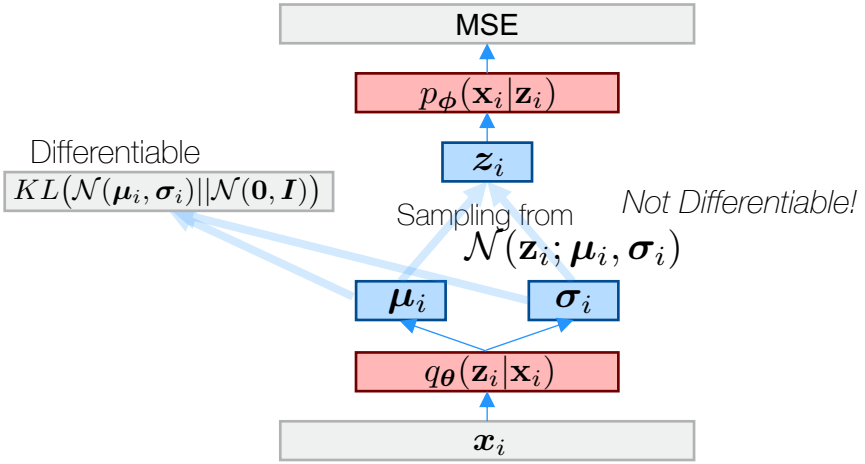
$$\mathcal{L}_i(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\mathbb{E}_{q_{\boldsymbol{\theta}}(\mathbf{z}_i|\boldsymbol{x}_i)}[\log p_{\boldsymbol{\phi}}(\mathbf{x}_i|\mathbf{z}_i)] + KL(q_{\boldsymbol{\theta}}(\mathbf{z}_i|\boldsymbol{x}_i)||p(\mathbf{z}))$$

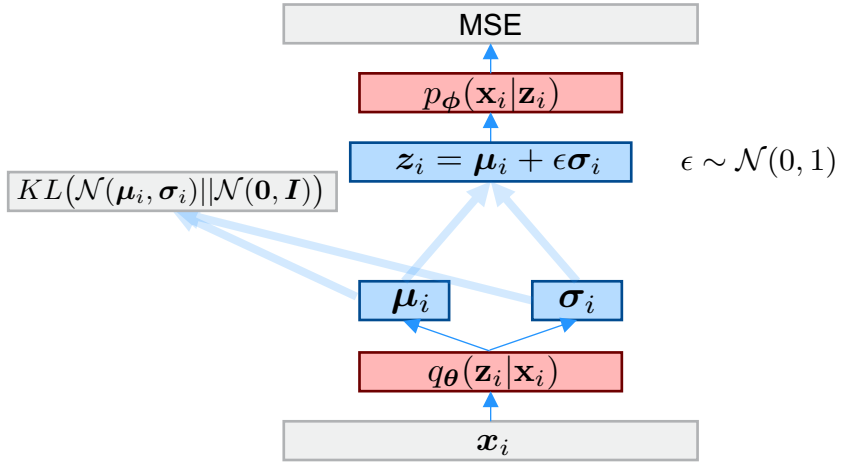
  - $q_{\boldsymbol{\theta}}(\mathbf{z}_i|\boldsymbol{x}_i)$: encoder network that produces the variational parameter $\boldsymbol{\lambda}_i = (\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$
  - $p_{\boldsymbol{\phi}}(\boldsymbol{x}_i|\mathbf{z}_i)$: decoder network that produces the distribution over data samples
  - $\boldsymbol{\theta}, \boldsymbol{\phi}$ are the network parameters you learn via SGD

Backpropagation on VAE

- Reparameterization

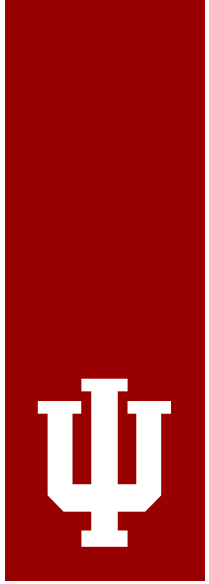
- How do we minimize $-ELBO_i(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\mathbb{E}_{q_{\boldsymbol{\theta}}(\mathbf{z}_i|\mathbf{x}_i)}[\log p_{\boldsymbol{\phi}}(\mathbf{x}_i|\mathbf{z}_i)] + KL\big(q_{\boldsymbol{\theta}}(\mathbf{z}_i|\mathbf{x}_i)||p(\mathbf{z})\big)$?





Reading

- Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114* (2013). https://arxiv.org/pdf/1312.6114.pdf
- Blei, David M.; Kucukelbir, Alp; McAuliffe, Jon D. “Variational Inference: A Review for Statisticians,” Journal of the American Statistical Association, Vol. 112 , Iss. 518, 2017 https://arxiv.org/abs/1601.00670
- Carl Doersch, “Tutorial on Variational Autoencoders,” https://arxiv.org/abs/1606.05908
- http://kvfrans.com/variational-autoencoders-explained/
- https://jaan.io/what-is-variational-autoencoder-vae-tutorial/
- https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf
- https://nips2017creativity.github.io/doc/Hierarchical_Variational_Autoencoders_for_Music.pdf

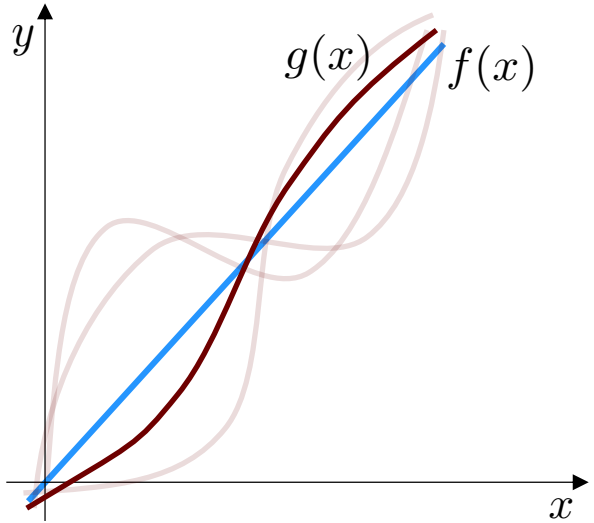


Thank You!

Variational?

- Calculus of variations

- First of all, you don't have to know this..



- It may look like a curve fitting problem, but it's different
  - Because it's an optimization over functions

$$\arg\max_{g \in \mathcal{G}} \mathcal{E}(g(x)||f(x))$$