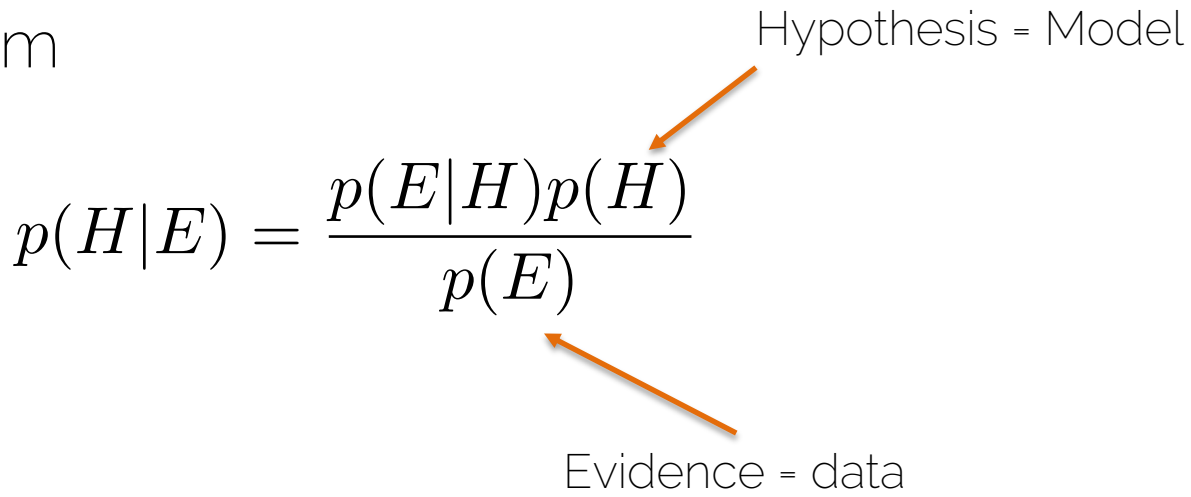
Bayesian Deep Learning

Going full Bayesian

- Bayes = Probabilities

- Bayes Theorem



Going full Bayesian

- Start with a prior on the model parameters $p(\theta)$
- Choose a statistical model $p(x|\theta)$

data

- Use data to refine my prior, i.e., compute the posterior

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

No dependence on parameters

Going full Bayesian

- Start with a prior on the model parameters $p(\theta)$
- Choose a statistical model $p(x|\theta)$

data

- Use data to refine my prior, i.e., compute the posterior

$$p(\theta|x) = p(x|\theta)p(\theta)$$

posterior likelihood prior

Going full Bayesian

- 1. Learning: Computing the posterior
  - Finding a point estimate (MAP) → what we have been doing so far!

$$p(\theta|x) = p(x|\theta)p(\theta)$$

  - Finding a probability distribution of θ

This lecture

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

What have we learned so far?

- Advantages of Deep Learning models
  - Very expressive models
  - Good for tasks such as classification, regression, sequence prediction
  - Modular structure, efficient training, many tools
  - Scales well with large amounts of data
- But we have also disadvantages…
  - "Black-box" feeling
  - We cannot judge how "confident" the model is about a decision

Modeling uncertainty

- Wish list:
  - We want to know what our models know and what they do not know

Modeling uncertainty

- Example: I have built a dog breed classifier

Bulldog

German sheperd

Chihuaha

What answer will my NN give?

Modeling uncertainty

- Example: I have built a dog breed classifier

Bulldog

German sheperd

Chihuaha

I would rather get as an answer that my model is not certain about the type of dog breed

Modeling uncertainty

- Wish list:
  - We want to know what our models know and what they do not know

- Why do we care?
  - Decision making
  - Learning from limited, noisy, and missing data
  - Insights on why a model failed

Modeling uncertainty

- Finding the posterior
  - Finding a point estimate (MAP) → what we have been doing so far!
  - Finding a probability distribution of θ

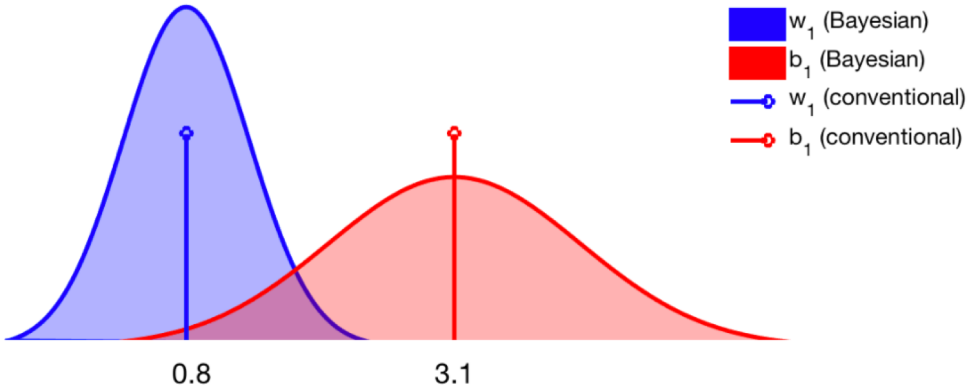


Image: https://medium.com/@joeDiHare/deep-bayesian-neural-networks-952763a9537

Modeling uncertainty

- We can sample many times from the distribution and see how this affects our model's predictions
- If predictions are consistent = model is confident

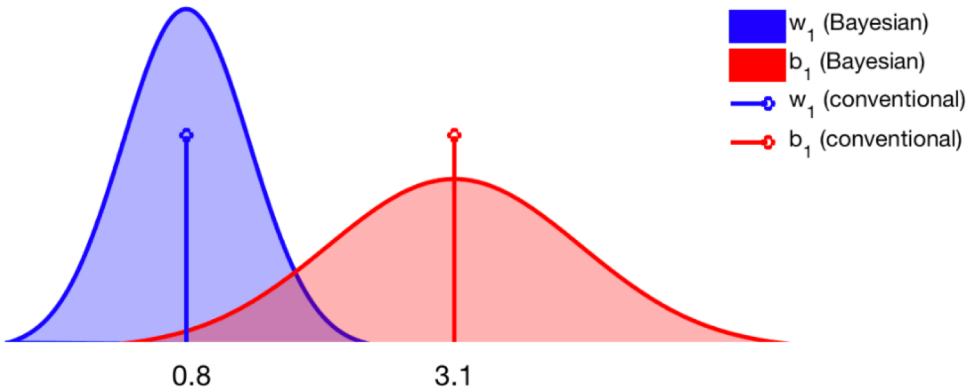
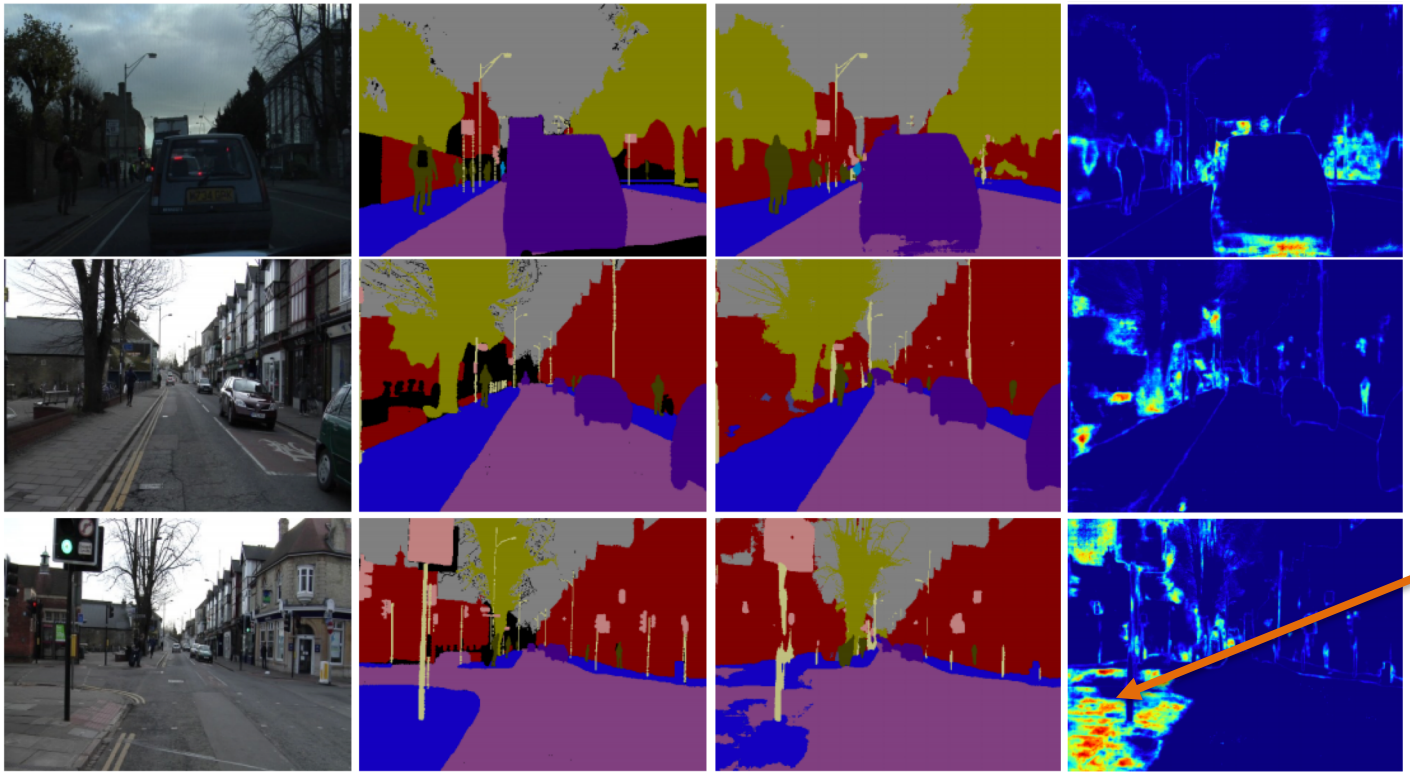


Image: https://medium.com/@joeDiHare/deep-bayesian-neural-networks-952763a9537

Modeling uncertainty



I am not really sure

Kendal & Gal. "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" NIPS 2016

Why?


NOVEMBER 6
SHELTON, WA
Shannon Adams via Storyful
AMHQ
SALMON SWIM ACROSS FLOODED STREET

How do we get the posterior?

- Compute the posterior over the weights

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

- Probability of observing our data under all possible model parameters

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{\int_\theta p(x|\theta)p(\theta)d\theta}$$

How do we compute this?

How do we get the posterior?

- How do we compute this?

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{\int_\theta p(x|\theta)p(\theta)d\theta}$$

- Denominator = we cannot compute all possible combinations

- Two ways to compute the approximation of the posterior:

Markov Chain Monte Carlo

Variational Inference

How do we get the posterior?

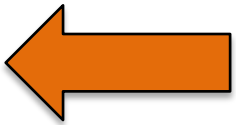
- Markov Chain Monte Carlo (MCMC)
  - A chain of samples

SLOW

$$\theta_t \rightarrow \theta_{t+1} \rightarrow \theta_{t+2} \quad \dots$$

that converge to $p(\theta|x)$

- Variational Inference
  - Find an approximation $q(\theta)$ that.

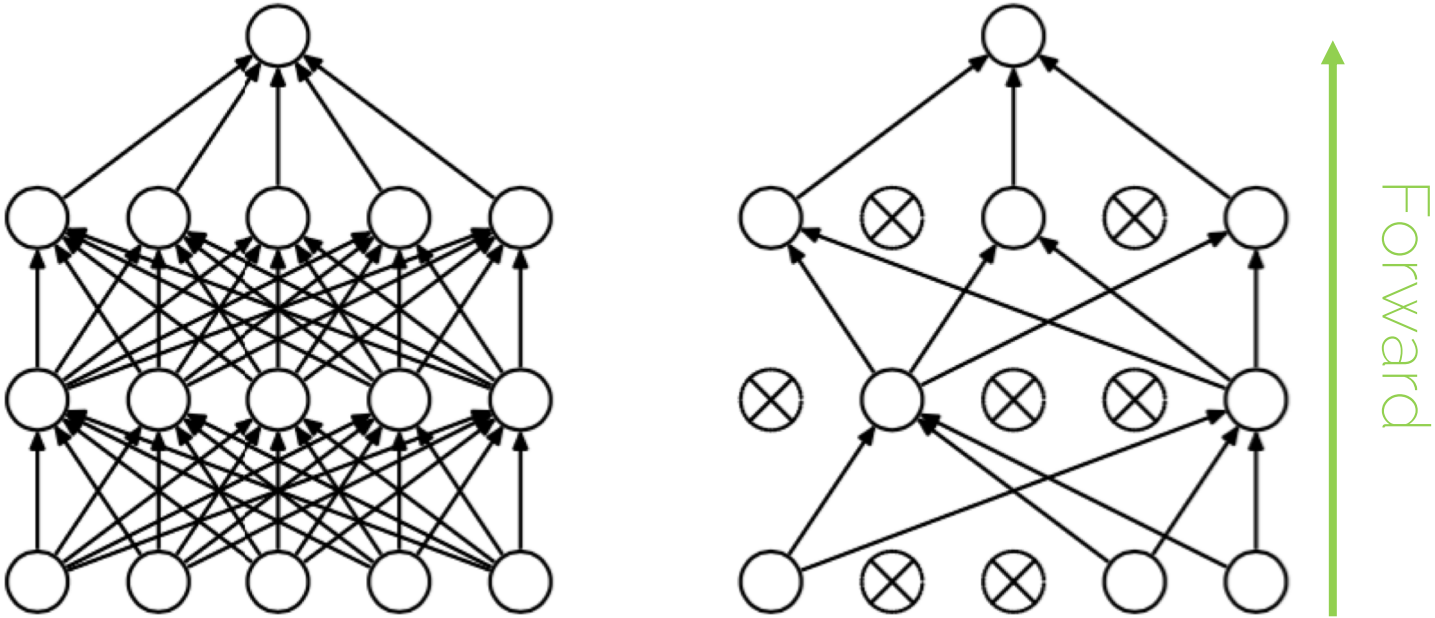


$$\arg\min KL(q(\theta)||p(\theta|x))$$

Dropout for Bayesian Inference

Recall: Dropout

- Disable a random set of neurons (typically 50%)



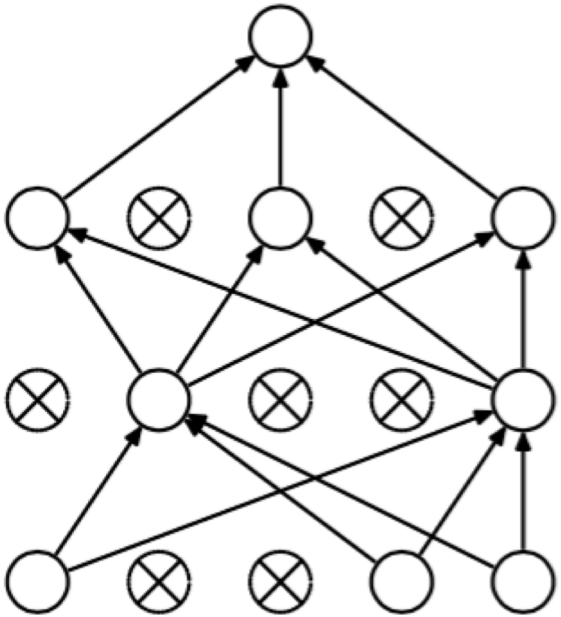
(a) Standard Neural Net

(b) After applying dropout.

Srivastava 2014

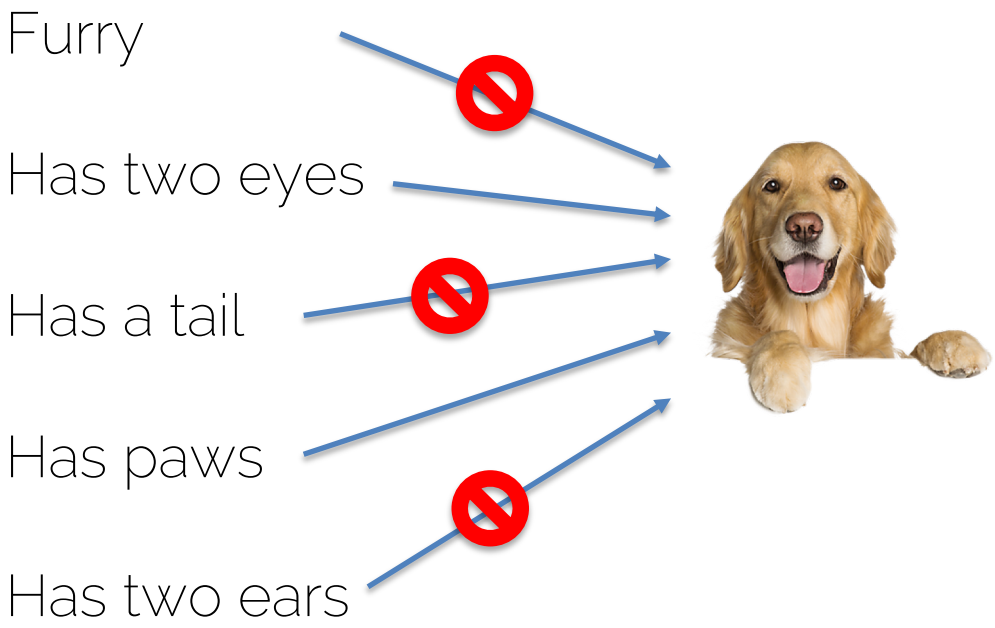
Recall: Dropout

- Using half the network = half capacity



(b) After applying dropout.

Redundant representations



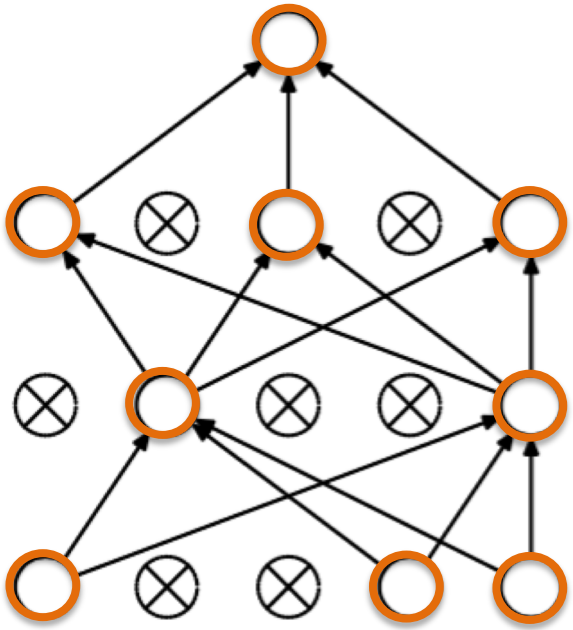
Recall: Dropout

- Using half the network = half capacity
  - Redundant representations
  - Base your scores on more features

- Consider it as model ensemble

Recall: Dropout

- Two models in one



(b) After applying dropout.

Model 1

Model 2

MC dropout

- Variational Inference
  - Find an approximation $q(\theta)$ that

$$\arg\min KL(q(\theta)||p(\theta|x))$$

- Dropout training
  - The variational distribution is from a Bernoulli distribution (where the states are "on" and "off")

Y Gal, Z Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning", ICML 2016

MC dropout

- 1. Train a model with dropout before every weight layer
- 2. Apply dropout at **test** time
  - Sampling is done in a Monte Carlo fashion, hence the name Monte Carlo dropout

Y Gal, Z Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning", ICML 2016

MC dropout

- Sampling is done in a Monte Carlo fashion, e.g.,

$$p(y = c|x) \approx \frac{1}{T} \sum_{t=1}^{T} Softmax(f_{\hat{\theta}_t}(x))$$

classification

NN

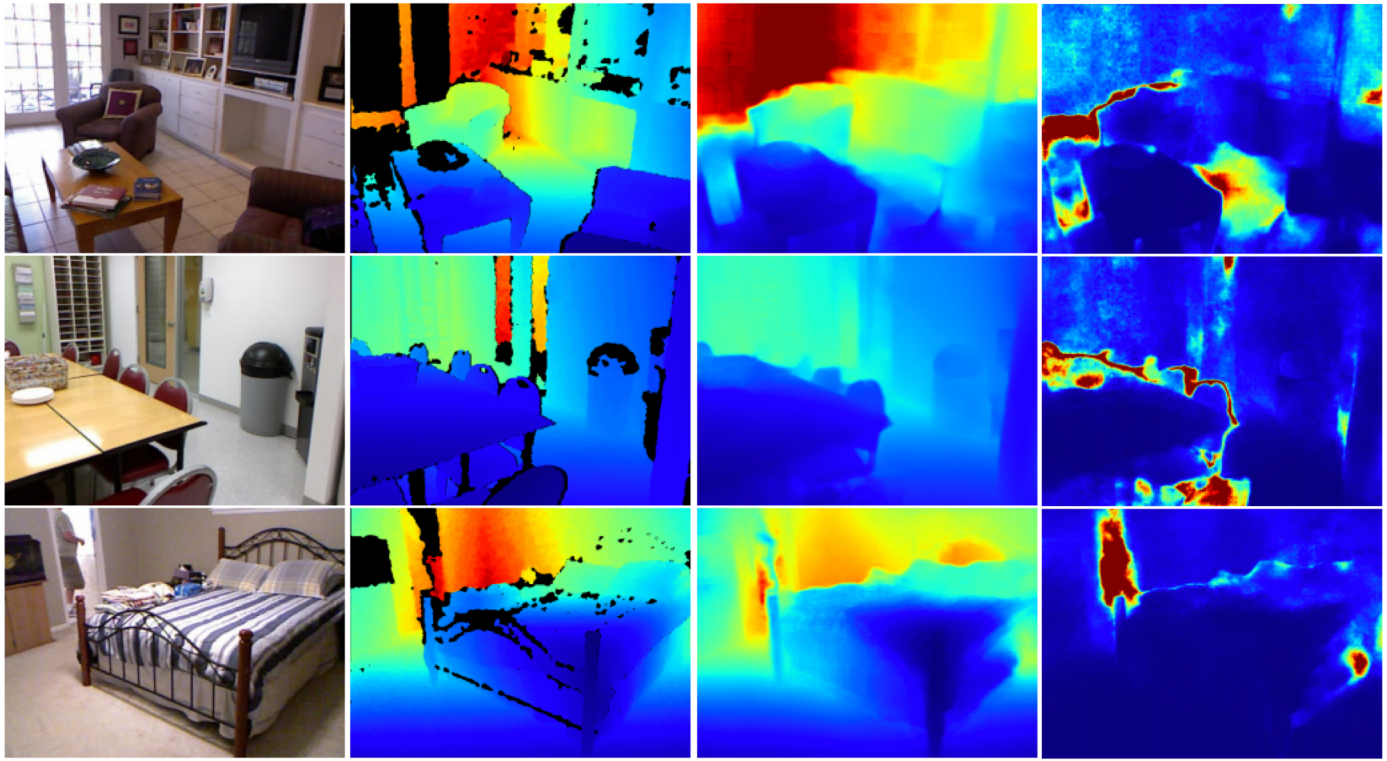
Parameter sampling

where $\hat{\theta}_t \sim q(\theta)$

and $q(\theta)$ is the dropout distribution

Y Gal, Z Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning", ICML 2016

Measure your model's uncertainty

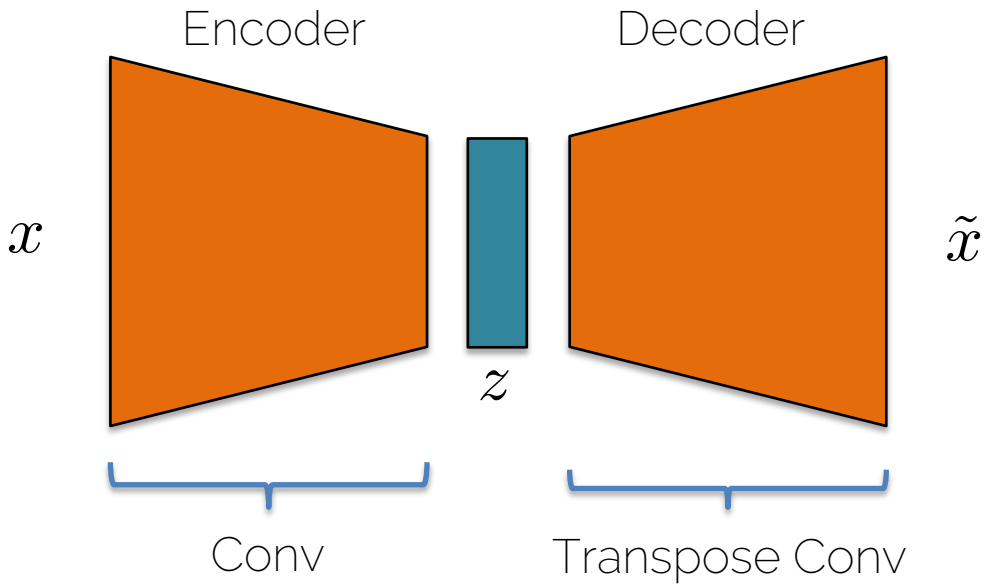


Kendal & Gal. "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" NIPS 2016

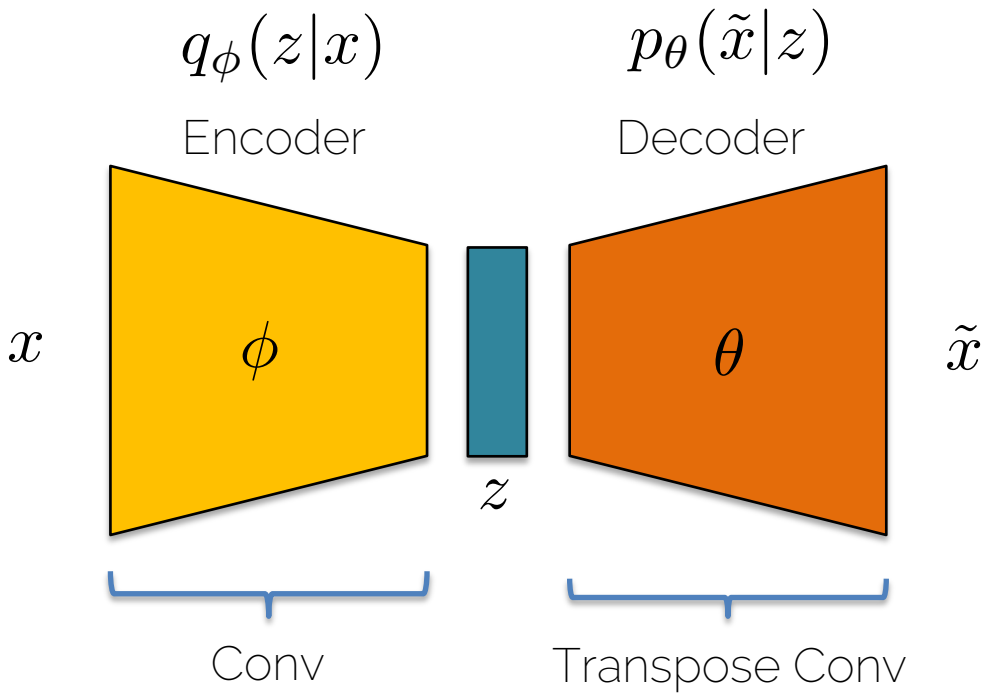
Variational Autoencoders

Recall: Autoencoders

- Encode the input into a representation (bottleneck) and reconstruct it with the decoder

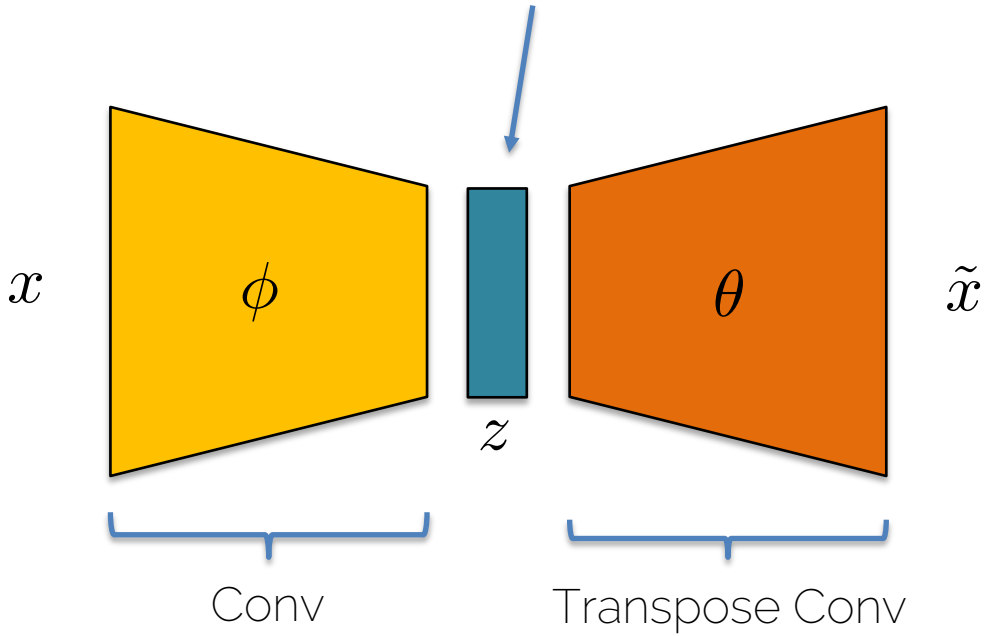


Variational Autoencoder



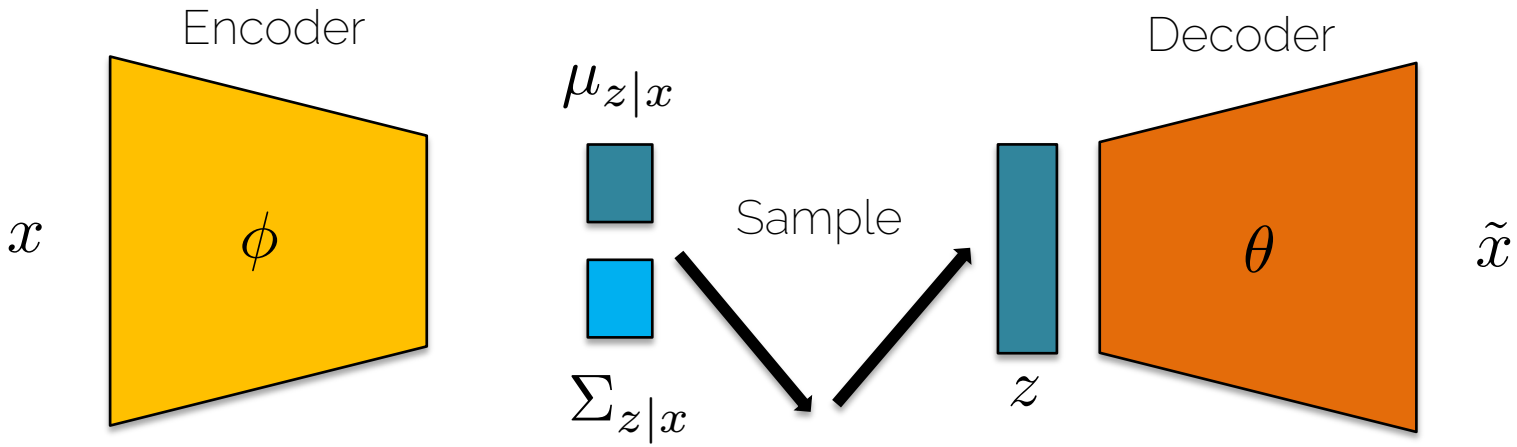
Variational Autoencoder

Goal: Sample from the latent distribution to generate new outputs!



Variational Autoencoder

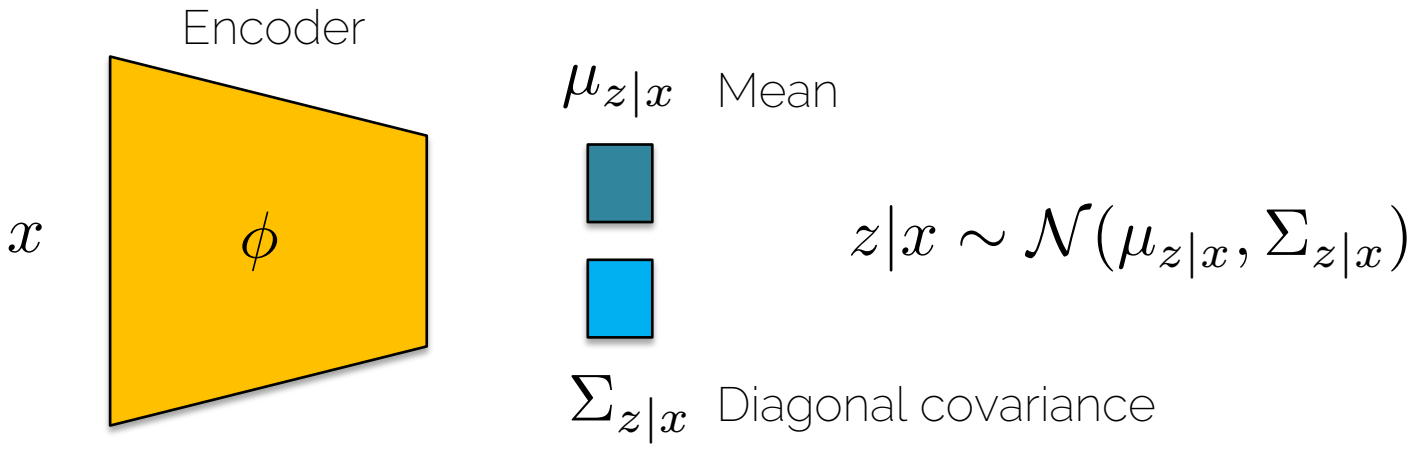
- Latent space is now a distribution
- Specifically it is a Gaussian



$$z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$$

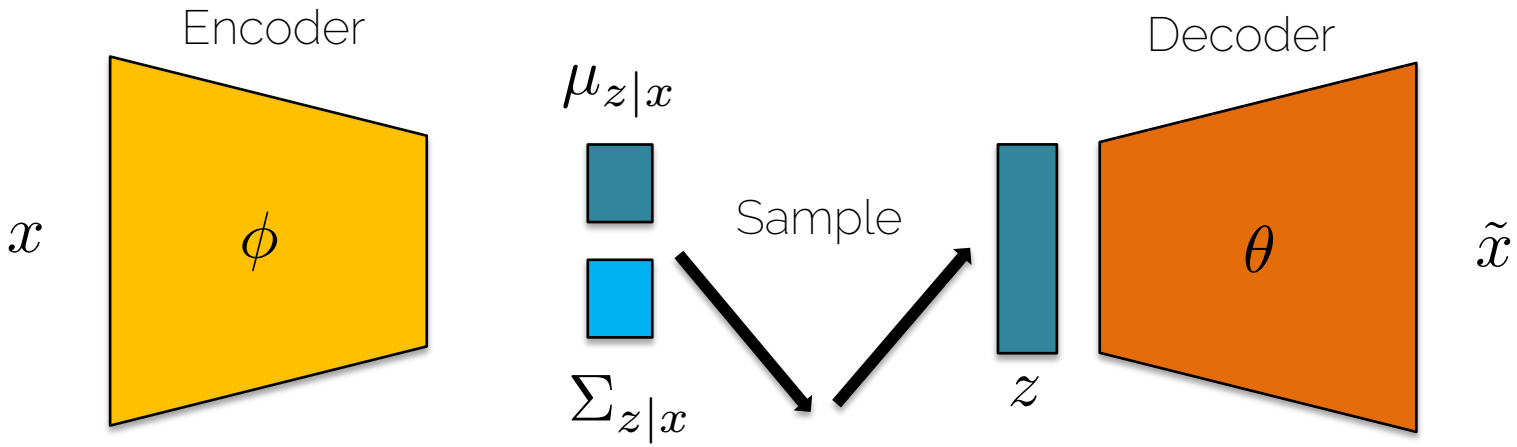
Variational Autoencoder

- Latent space is now a distribution
- Specifically it is a Gaussian



Variational Autoencoder

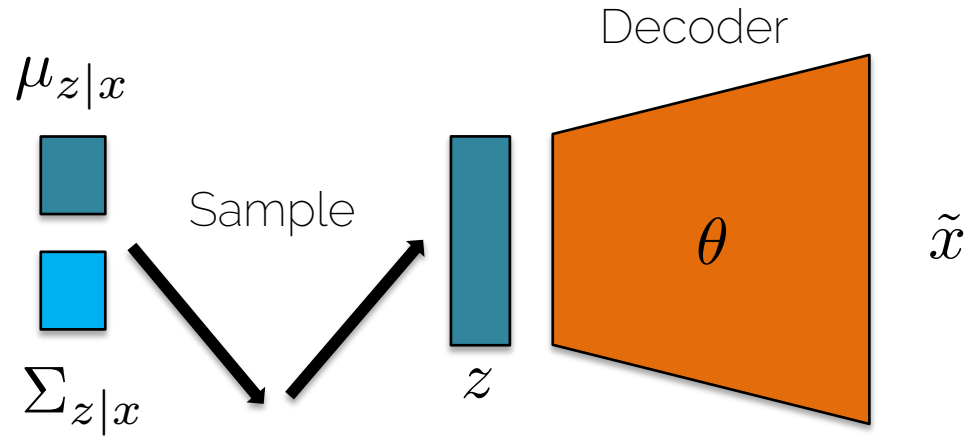
- Training



$$z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$$

Variational Autoencoder

- Test: sampling from the latent space



$$z|x \sim \mathcal{N}(\mu_{z|x}, \Sigma_{z|x})$$

VAE: training

- Back to the Bayesian view for training

$$p_\theta(x) = \int_z p_\theta(x|z)p_\theta(z)dz$$

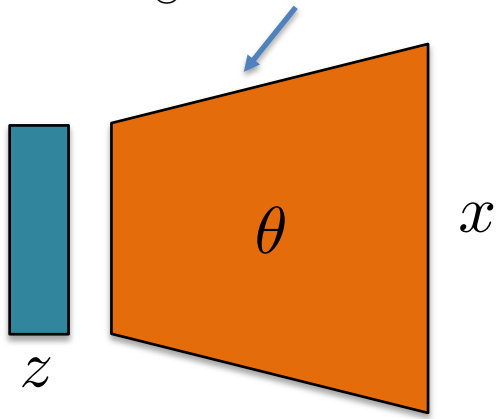
Intractable to compute the output for every z

Decoder (Neural Network)

Prior = Gaussian

$p_\theta(z)$

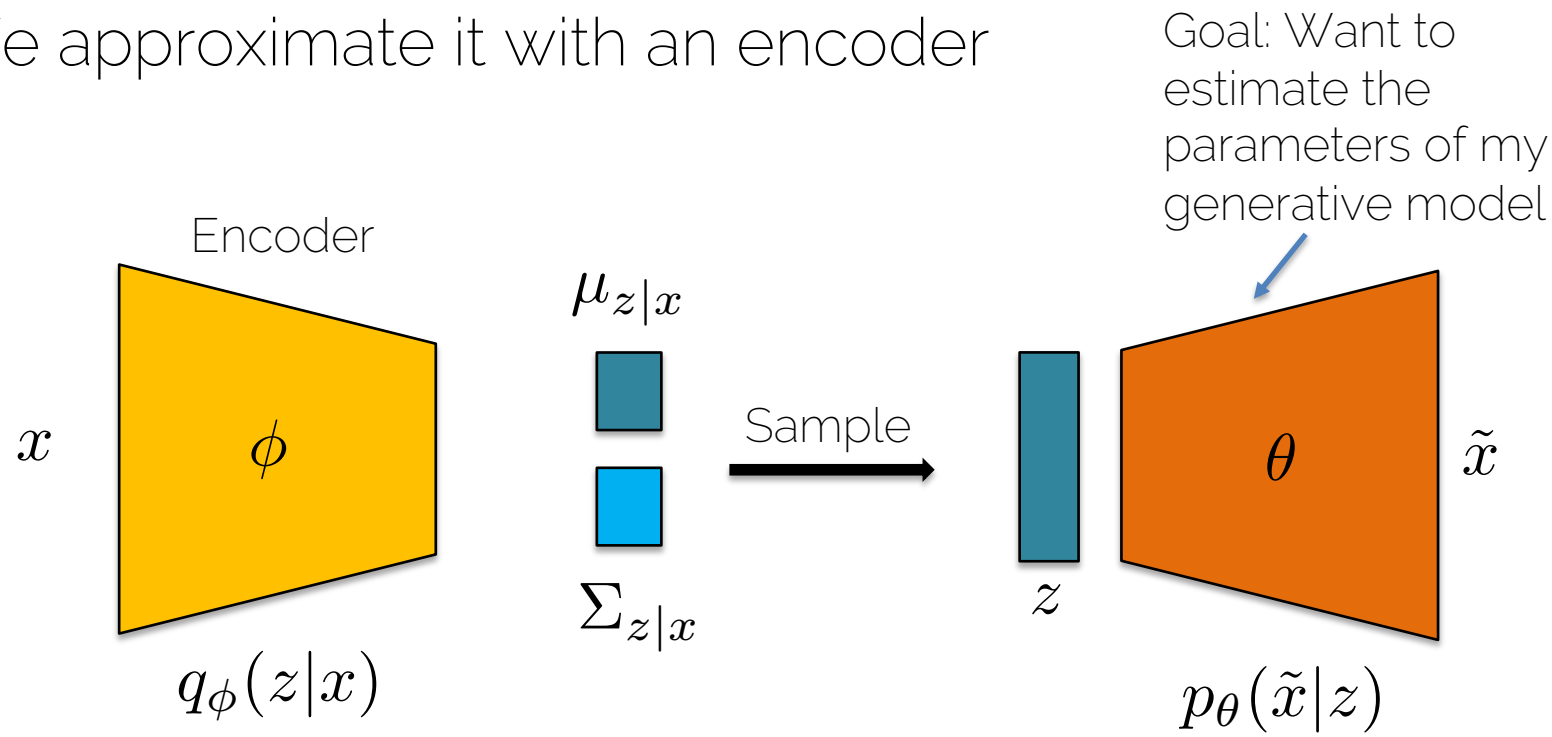
Goal: Want to estimate the parameters of my generative model



$p_\theta(x|z)$

VAE: training

- We approximate it with an encoder



VAE: loss function

- Loss function for a data point x_i

$$\log(p_\theta(x_i)) = \boldsymbol{E}_{z\sim q_\phi(z|x_i)}[\log(p_\theta(x_i))]$$

I draw samples of the latent variable z from my encoder

VAE: loss function

- Loss function for a data point x_i

$$\log(p_\theta(x_i)) = \boldsymbol{E}_{z\sim q_\phi(z|x_i)}[\log(p_\theta(x_i))]$$

$$= \boldsymbol{E}_{z\sim q_\phi(z|x_i)}\left[\log \frac{p_\theta(x_i|z)p_\theta(z)}{p_\theta(z|x_i)}\right]$$

Bayes Rule

Recall:

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}$$

Using the latent variable, which will become useful to simplify the expressions later according to our AE formulation

VAE: loss function

- Loss function for a data point x_i

$$\log(p_\theta(x_i)) = \boldsymbol{E}_{z\sim q_\phi(z|x_i)}[\log(p_\theta(x_i))]$$

$$= \boldsymbol{E}_{z\sim q_\phi(z|x_i)}\left[\log \frac{p_\theta(x_i|z)p_\theta(z)}{p_\theta(z|x_i)}\right]$$

$$= \boldsymbol{E}_z\left[\log \frac{p_\theta(x_i|z)p_\theta(z)}{p_\theta(z|x_i)}\frac{q_\phi(z|x_i)}{q_\phi(z|x_i)}\right]$$

Just a constant

VAE: loss function

- Loss function for a data point x_i

$$\log(p_\theta(x_i)) = \boldsymbol{E}_z \left[\log \frac{p_\theta(x_i|z) p_\theta(z) q_\phi(z|x_i)}{p_\theta(z|x_i) q_\phi(z|x_i)} \right]$$

$$= \boldsymbol{E}_z \left[\log p_\theta(x_i|z) \right] - \boldsymbol{E}_z \left[\log \frac{q_\phi(z|x_i)}{p_\theta(z)} \right] + \boldsymbol{E}_z \left[\log \frac{q_\phi(z|x_i)}{p_\theta(z|x_i)} \right]$$

Apply the logarithm and group as needed

VAE: loss function

- Loss function for a data point x_i

$$= \boldsymbol{E}_z \left[\log p_\theta(x_i|z)\right] - \boldsymbol{E}_z \left[\log \frac{q_\phi(z|x_i)}{p_\theta(z)}\right] + \boldsymbol{E}_z \left[\log \frac{q_\phi(z|x_i)}{p_\theta(z|x_i)}\right]$$

Kullback-Leibler Divergences to measure how similar two distributions are

VAE: loss function

- Loss function for a data point x_i

$$= \boldsymbol{E}_z \left[\log p_\theta(x_i|z)\right] - \boldsymbol{E}_z \left[\log \frac{q_\phi(z|x_i)}{p_\theta(z)}\right] + \boldsymbol{E}_z \left[\log \frac{q_\phi(z|x_i)}{p_\theta(z|x_i)}\right]$$

$$= \boldsymbol{E}_z \left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$

Kullback-Leibler Divergences

VAE: loss function

- Loss function for a data point x_i

$$= \boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$

Reconstruction loss (how well does my decoder reconstruct a data point given the latent vector z). We need to sample from z.

Measures how good my latent distribution is with respect to my Gaussian prior

I still cannot express the shape of the distribution. But I know

$$\geq 0$$

VAE: loss function

- Loss function for a data point x_i

$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$

Loss function (lower bound)

$$\mathcal{L}(x_i, \phi, \theta)$$

$$\geq 0$$

$$\log(p(x_i)) \geq \mathcal{L}(x_i, \phi, \theta)$$

VAE: loss function

- Loss function for a data point x_i

$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$

Loss function (lower bound)

$$\mathcal{L}(x_i, \phi, \theta)$$

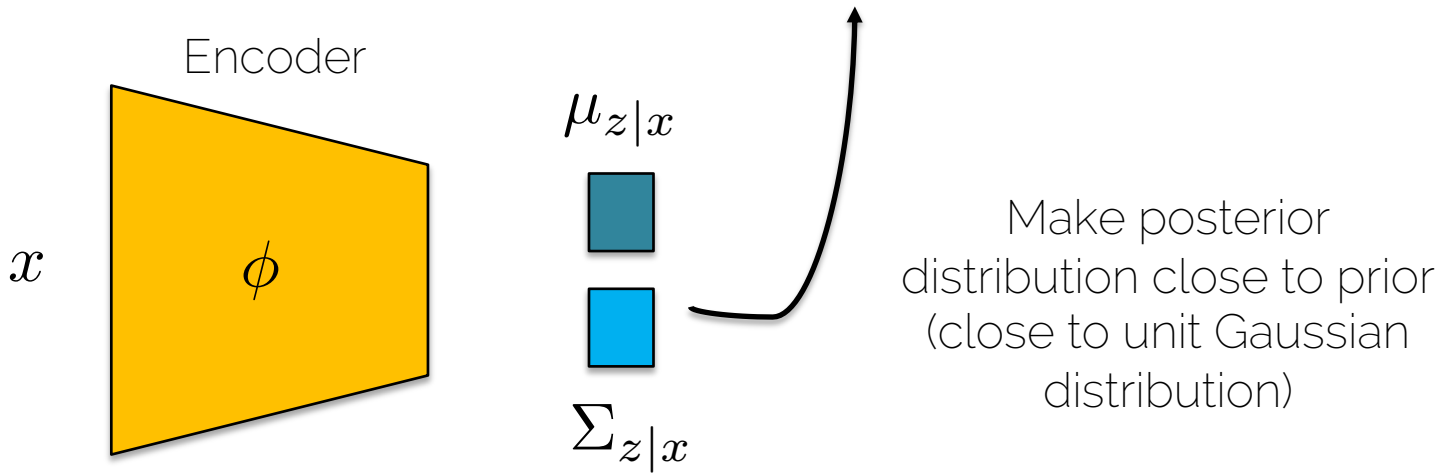
$$\geq 0$$

- Optimize $\phi^*, \theta^* = \arg\max \sum_{i=1}^{N} \mathcal{L}(x_i, \phi, \theta)$

Variational Autoencoder

- Training

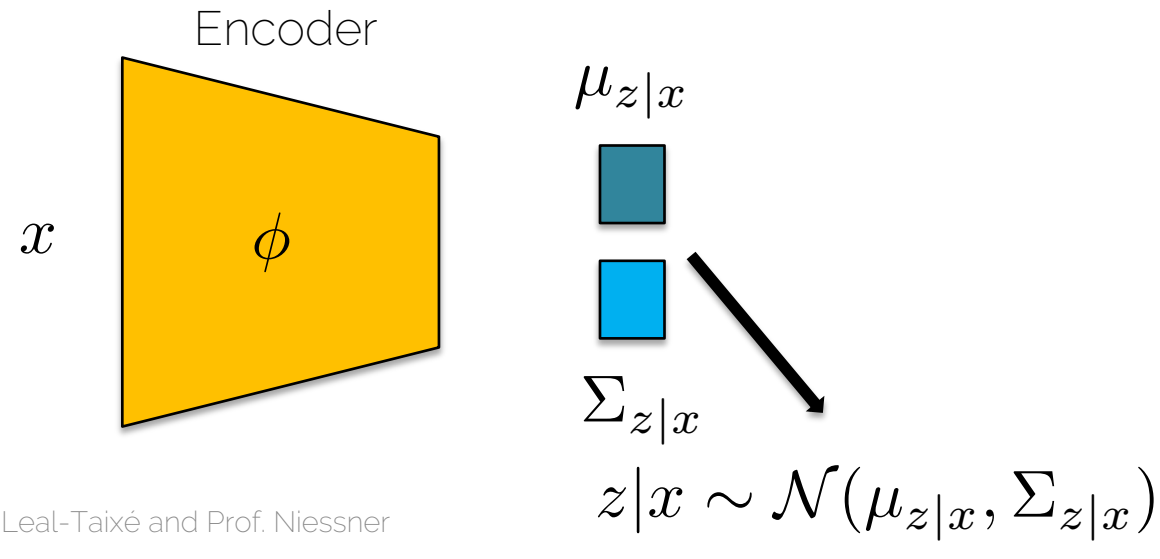
$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$



Variational Autoencoder

- Training

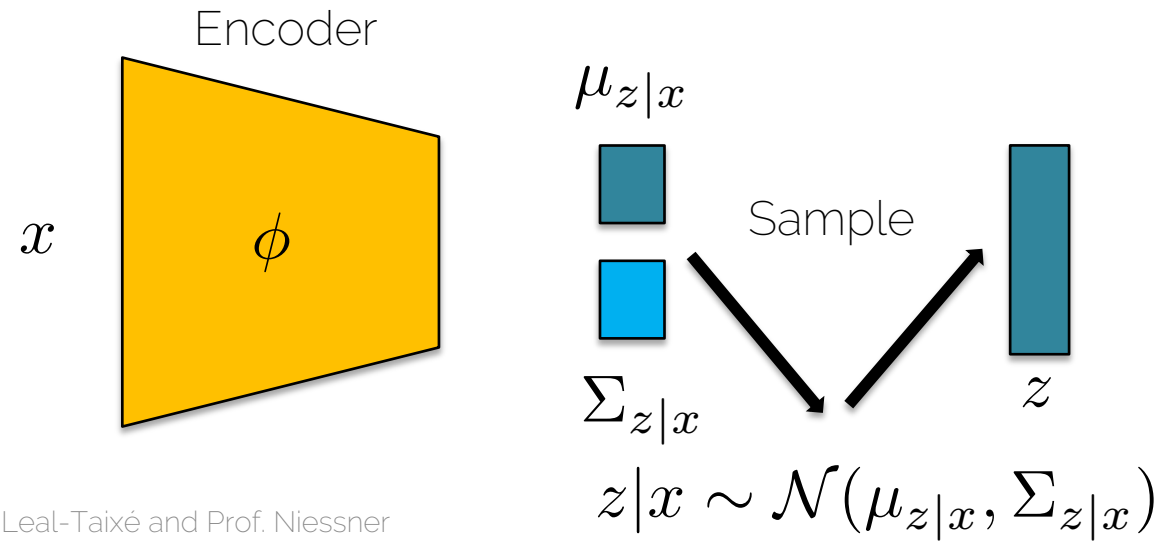
$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$



Variational Autoencoder

- Training

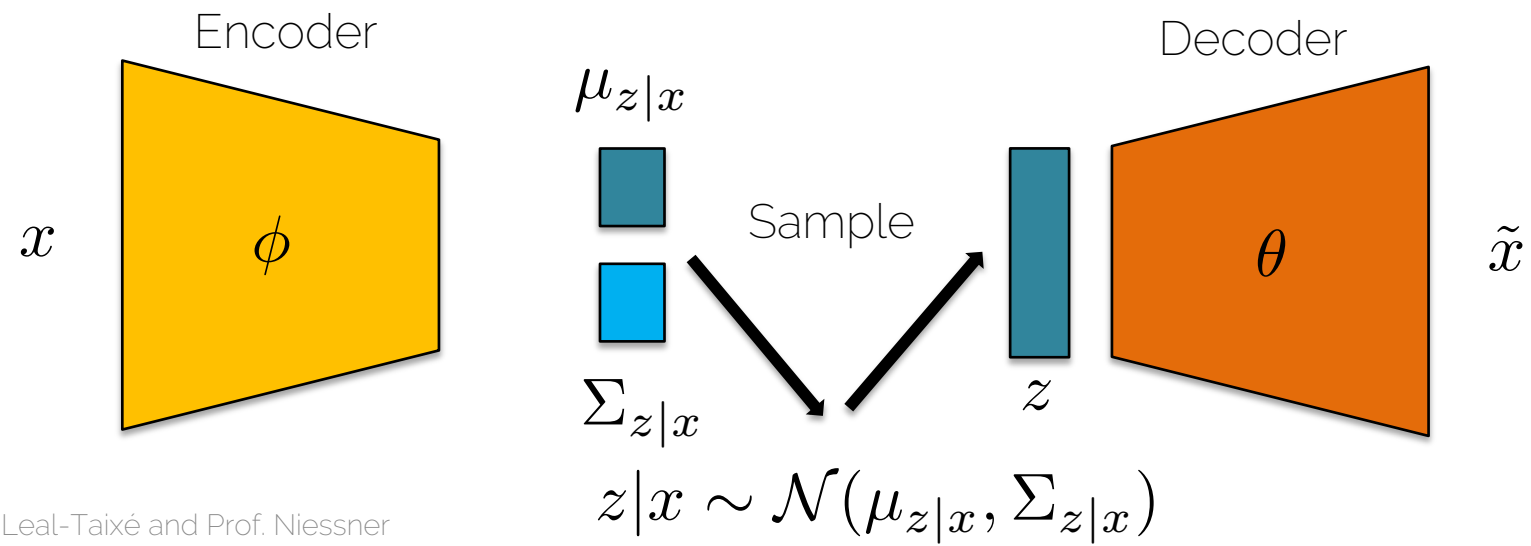
$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$



Variational Autoencoder

- Training

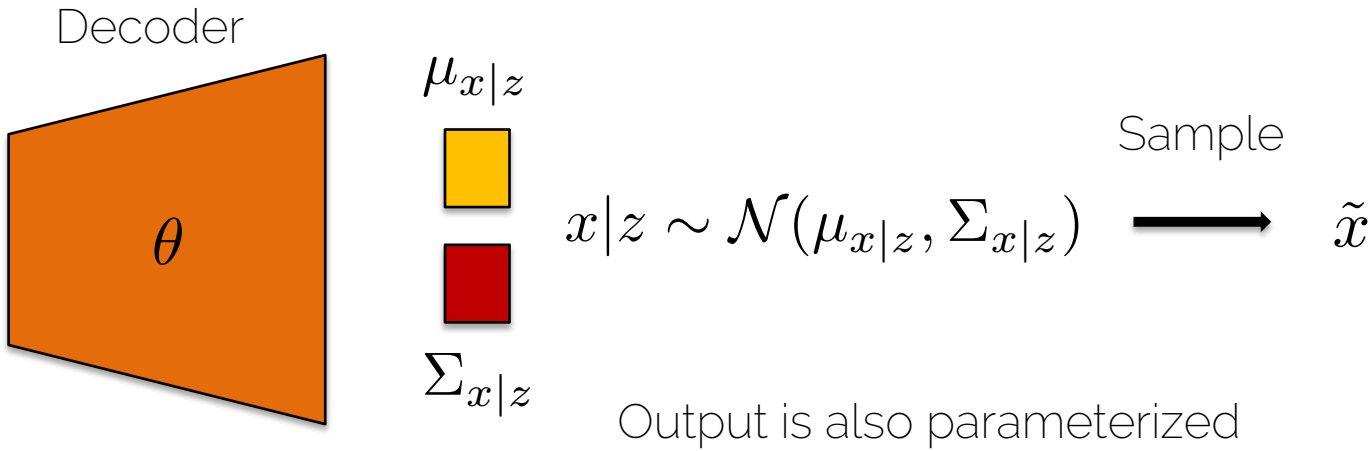
$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$



Variational Autoencoder

- Training

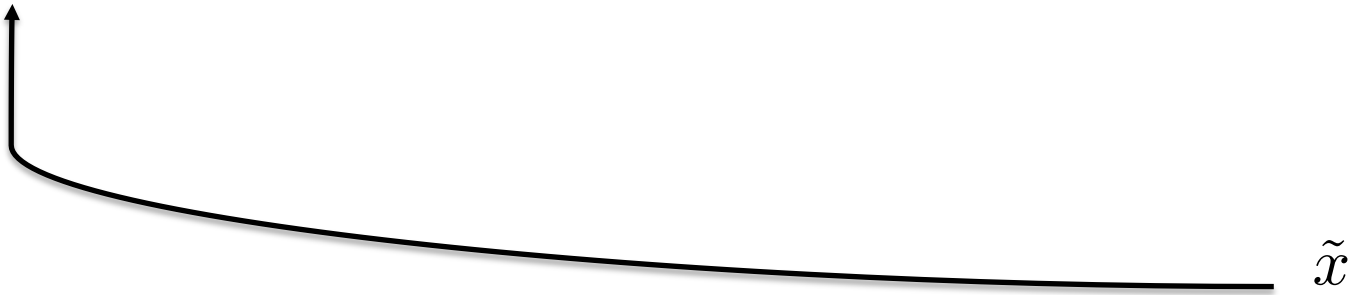
$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$



Variational Autoencoder

- Training

$$\boldsymbol{E}_z\left[\log p_\theta(x_i|z)\right] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$



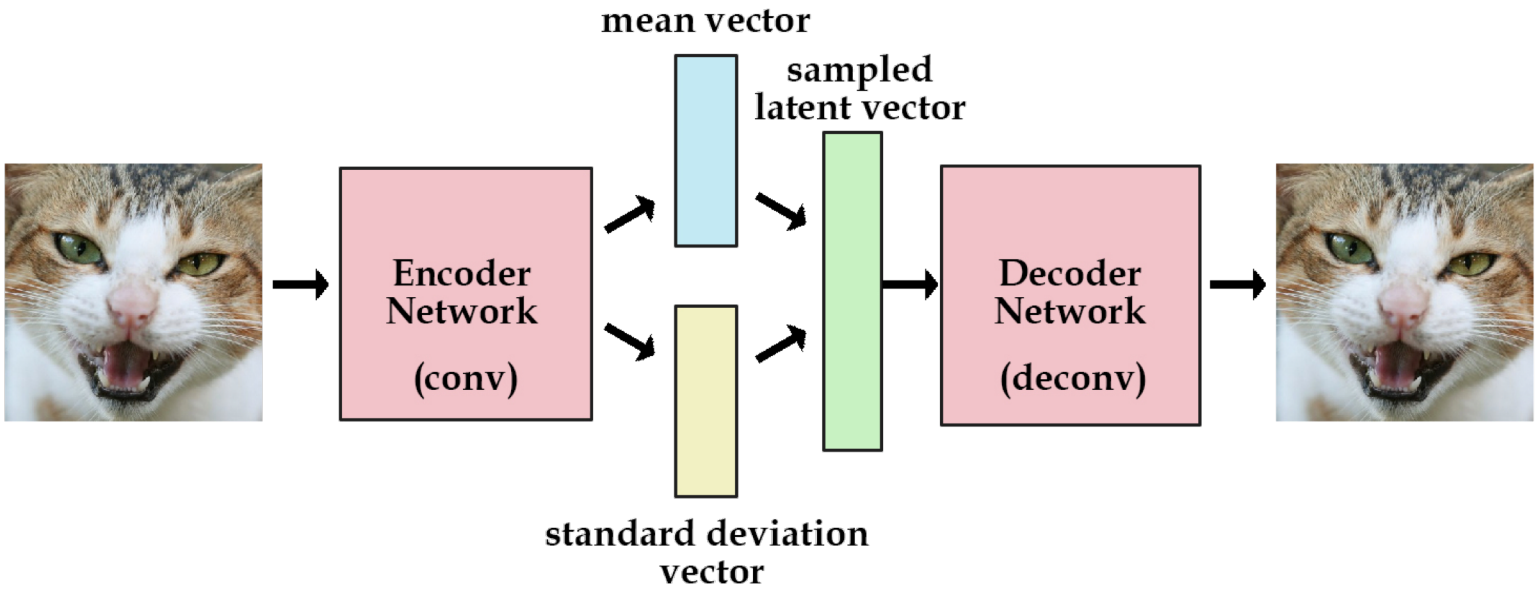
Maximize the likelihood of reconstructing the input

Variational Autoencoder

- For more details and mathematical derivation
- Reparameterization trick (expressing variables as Gaussians) that allows us to perform backpropagation
- Kingman and Welling. "Auto-Encoding Variational Bayes". ICLR 2014

How about generating data?

- Training as seen before

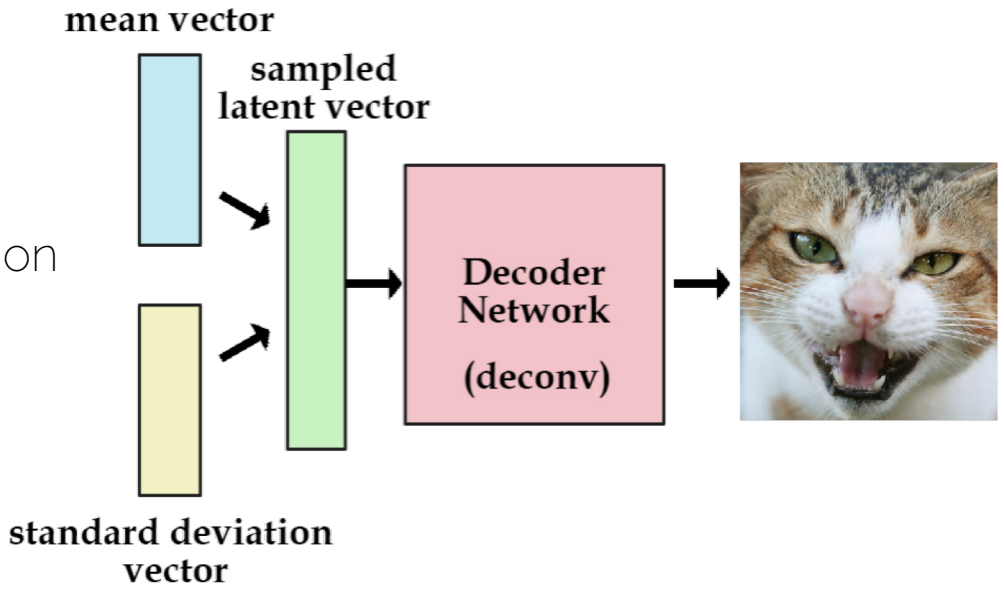


http://kvfrans.com/variational-autoencoders-explained/

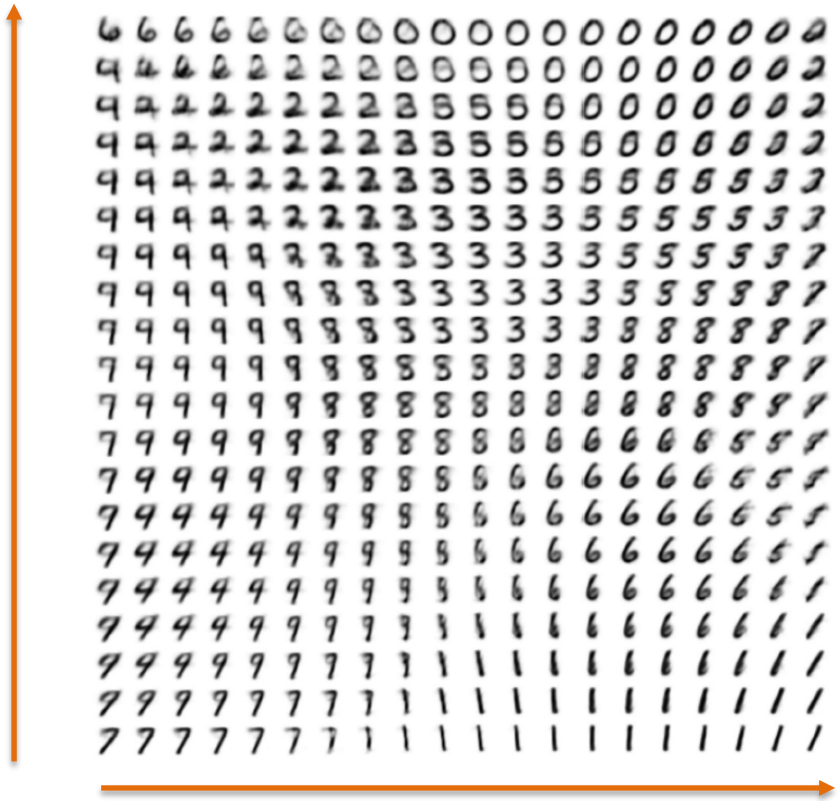
How about generating data?

- After training, generate random samples

Sample from the distribution
(e.g., unit Gaussian



Generating data



Each element of z encodes a different feature

Generating data

Degree of smile

Head pose

Autoencoder vs VAE

Autoencoder

Variational Autoencoder

Ground Truth

https://github.com/kvfrans/variational-autoencoder

Autoencoder Overview

- Autoencoders (AE)
  - Reconstruct input
  - Unsupervised learning
  - Latent space features are useful

- Variational Autoencoders (VAE)
  - Probability distribution in latent space (e.g., Gaussian)
  - Interpretable latent space (head pose, smile)
  - Sample from model to generate output

Next lectures

- More on Generative models

- This Wednesday 4th, first project presentations!

Other references

- Conditional Variational Autoencoders:
  - Sohn, Kihyuk, Honglak Lee, and Xinchen Yan. "Learning Structured Output Representation using Deep Conditional Generative Models." Advances in Neural Information Processing Systems. 2015.
  - Xinchen Yan, Jimei Yang, Kihyuk Sohn, Honglak Lee, Attribute2Image: Conditional Image Generation from Visual Attributes, ECCV, 2016 –

Other references

- Interesting read:
  - Jacob Walker, Carl Doersch, Abhinav Gupta, Martial Hebert, An Uncertain Future: Forecasting from Static Images using Variational Autoencoders, ECCV, 2016
  - Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, Ole Winther, Autoencoding beyond pixels using a learned similarity metric, ICML, 2016
  - Aditya Deshpande, Jiajun Lu, Mao-Chuang Yeh, David Forsyth, Learning Diverse Image Colorization, arXiv, 2016
  - Raymond Yeh, Ziwei Liu, Dan B Goldman, Aseem Agarwala, Semantic Facial Expression Editing using Autoencoded Flow, arXiv, 2016
  - Diederik P. Kingma, Danilo J. Rezende, Shakir Mohamed, Max Welling, Semi-Supervised Learning with Deep Generative Models, NIPS, 2014