10-708 Probabilistic Graphical Models

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Variational Autoencoders

Matt Gormley
Lecture 20
April 12, 2021

Reminders

- Quiz 2
  – Wed, Apr 14, during lecture time
- HW5 Recitation
  – Wed, Apr. 14 at 7pm
- Homework 5: Variational Inference
  – Out: Thu, Apr. 8
  – Due: Wed, Apr. 21 at 11:59pm
- Project Midway Milestones:
  – Midway Poster Session:
    Tue, Apr. 27 at 6:30pm – 8:30pm
  – Midway Executive Summary
    Due: Tue, Apr. 27 at 11:59pm
  – New requirement: must have baseline results

QUIZ 2 LOGISTICS

Quiz 2

- **Time / Location**
  - **Time:** In-Class Quiz
    Wed, Apr. 14 during lecture time
  - **Location:** The same Zoom meeting as lecture/recitation.
    Please arrive online early.
  - Please watch Piazza carefully for announcements.
- **Logistics**
  - Covered material: Lecture 9 – Lecture 15
    (and unavoidably some material from Lectures 1 – 8)
  - Format of questions:
    - Multiple choice
    - True / False (with justification)
    - Derivations
    - Short answers
    - Interpreting figures
    - Implementing algorithms on paper
    - Drawing
  - No electronic devices
  - You are allowed to **bring** one 8½ x 11 sheet of notes (front and back)

Quiz 2

- **Advice (for before the exam)**
  - Try out the Gradescope quiz-style interface in the "Fake Quiz" now available
- **Advice (for during the exam)**
  - Solve the easy problems first
    (e.g. multiple choice before derivations)
    - if a problem seems extremely complicated you're likely missing something
  - Don't leave any answer blank!
  - If you make an assumption, write it down
  - If you look at a question and don't know the answer:
    - we probably haven't told you the answer
    - but we've told you enough to work it out
    - imagine arguing for some answer and see if you like it

Topics for Quiz 1

- Graphical Model Representation
  - Directed GMs vs. Undirected GMs vs. Factor Graphs
  - Bayesian Networks vs. Markov Random Fields vs. Conditional Random Fields
- Graphical Model Learning
  - Fully observed Bayesian Network learning
  - Fully observed MRF learning
  - Fully observed CRF learning
  - Parameterization of a GM
  - Neural potential functions
- Exact Inference
  - Three inference problems: (1) marginals (2) partition function (3) most probably assignment
  - Variable Elimination
  - Belief Propagation (sum-product and max-product)

Topics for Quiz 2

- Learning for Structure Prediction
  - Structured Perceptron
  - Structured SVM
  - Neural network potentials
- (Approximate) MAP Inference
  - MAP Inference via MILP
  - MAP Inference via LP relaxation
- Approximate Inference by Sampling
  - Monte Carlo Methods
  - Gibbs Sampling
  - Metropolis-Hastings
  - Markov Chains and MCMC
- Parameter Estimation
  - Bayesian inference
  - Topic Modeling

Q&A

AUTOENCODERS

Unsupervised Pre-training

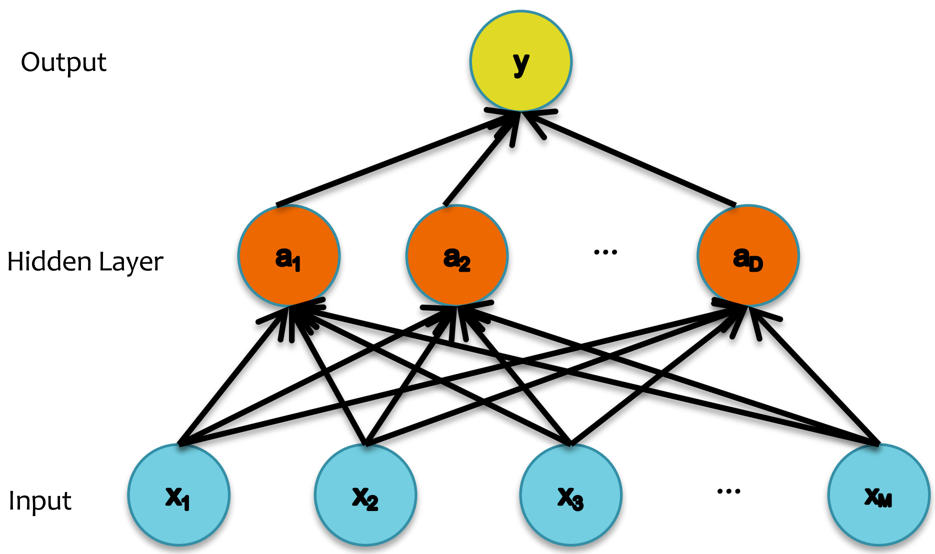
- **Idea: (Two Steps)**
  - Use supervised learning, but **pick a better starting point**
  - **Train each level** of the model in a **greedy** way

1. Unsupervised Pre-training
   - Use **unlabeled** data
   - Work bottom-up
     - Train hidden layer 1. Then fix its parameters.
     - Train hidden layer 2. Then fix its parameters.
     - …
     - Train hidden layer n. Then fix its parameters.
2. Supervised Fine-tuning
   - Use **labeled** data to train following "Idea #1"
   - Refine the features by backpropagation so that they become tuned to the end-task

Unsupervised Pre-training

Unsupervised pre-training of the first layer:

- What should it predict?
- What else do we observe?
- **The input!**

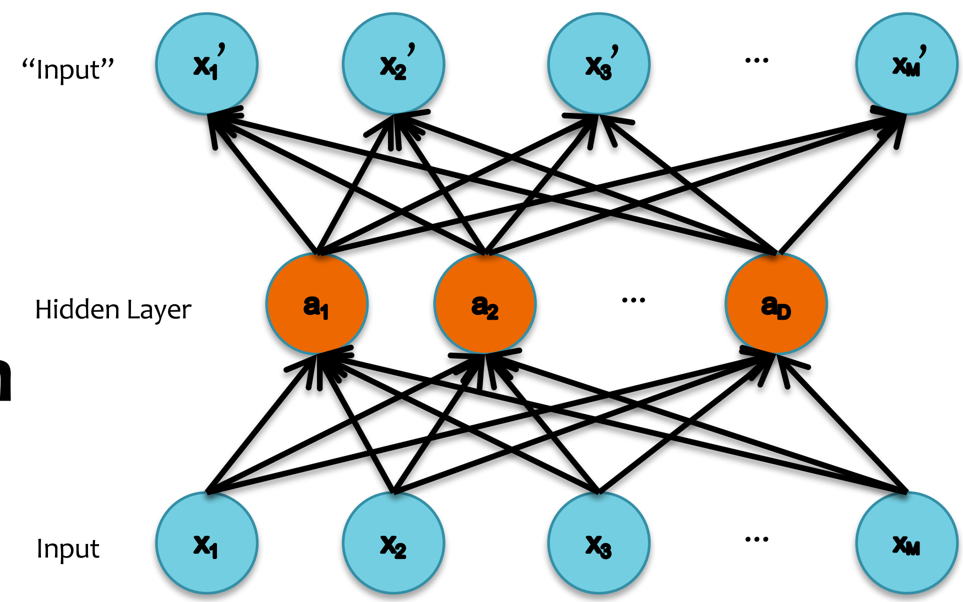


Auto-Encoders

Unsupervised pre-training of the first layer:

- What should it predict?
- What else do we observe?
- **The input!**

This topology defines an Auto-encoder.



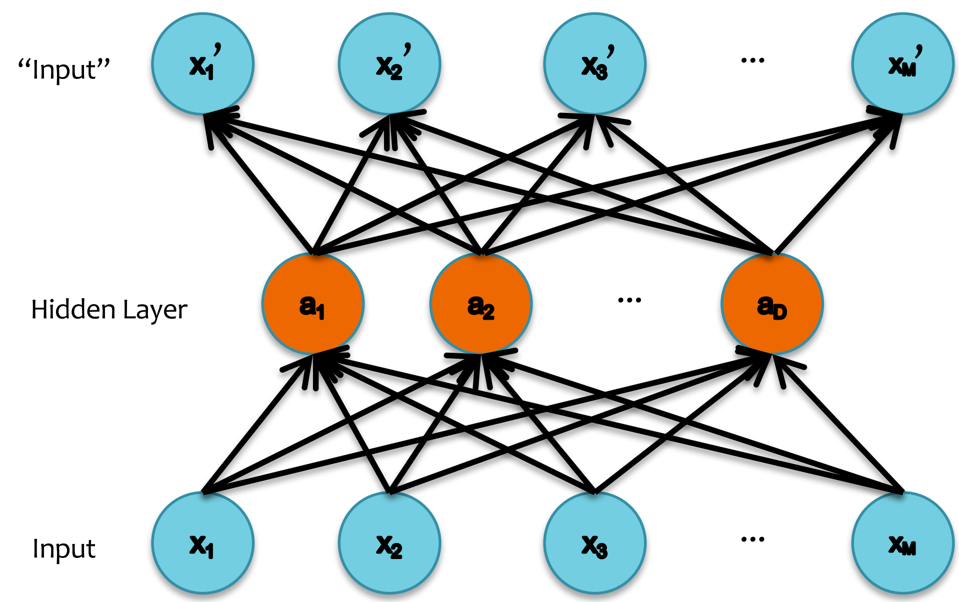
Auto-Encoders

Key idea: Encourage z to give small reconstruction error:

- x' is the *reconstruction* of x
- Loss = || x – DECODER(ENCODER(x)) ||^2
- Train with the same backpropagation algorithm for 2-layer Neural Networks with x_m as both input and output.

DECODER: x' = h(W'z)

ENCODER: z = h(Wx)

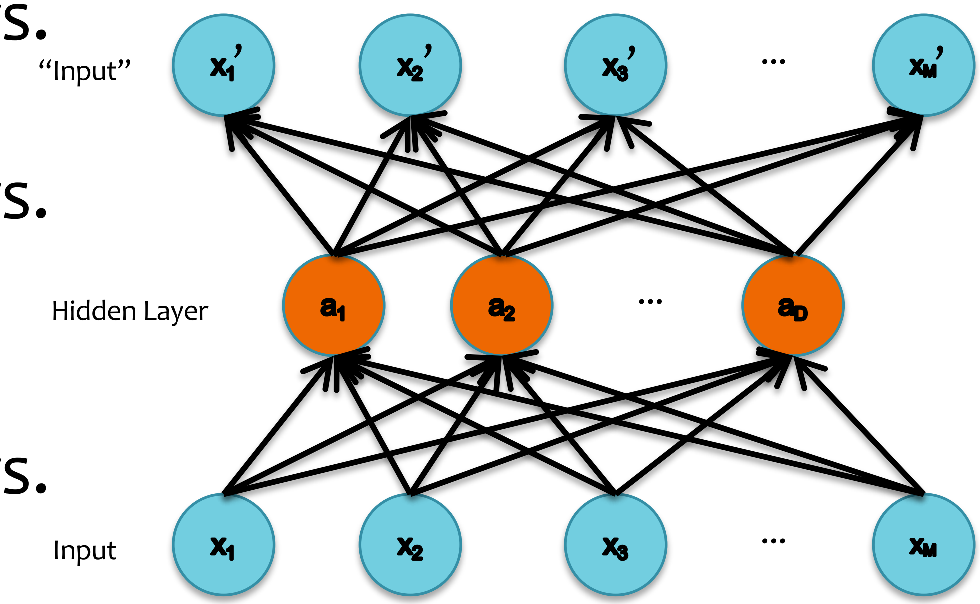


Slide adapted from Raman Arora

Unsupervised Pre-training

Unsupervised pre-training

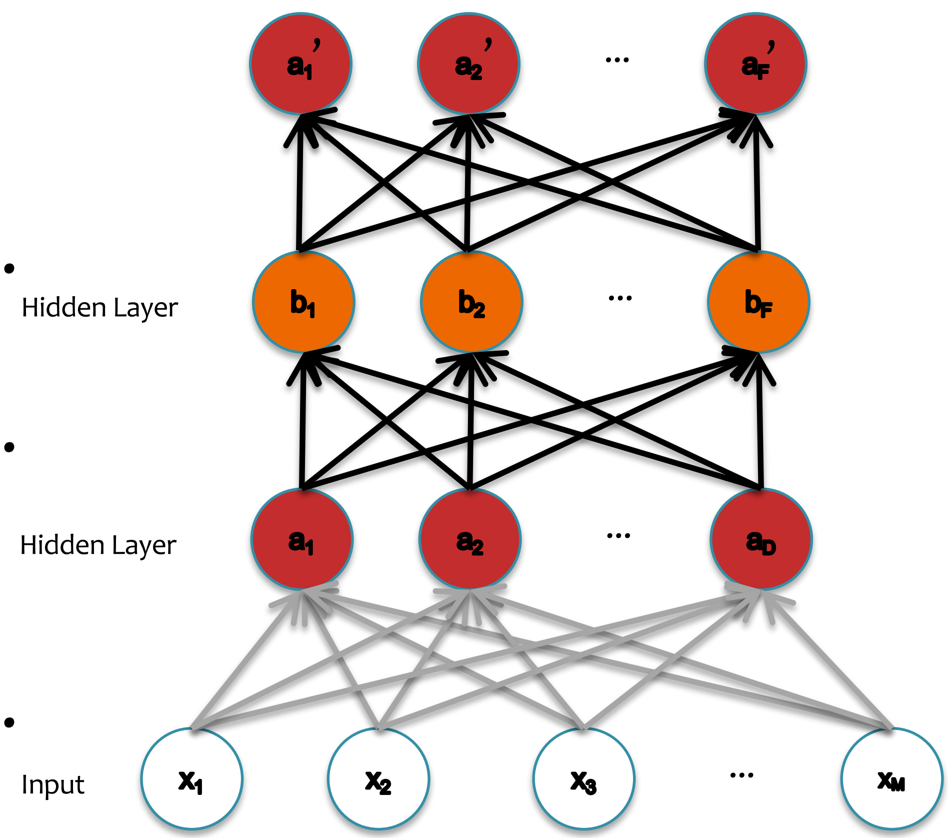
- Work bottom-up
  - Train hidden layer 1. Then fix its parameters.
  - Train hidden layer 2. Then fix its parameters.
  - …
  - Train hidden layer n. Then fix its parameters.



Unsupervised Pre-training

Unsupervised pre-training

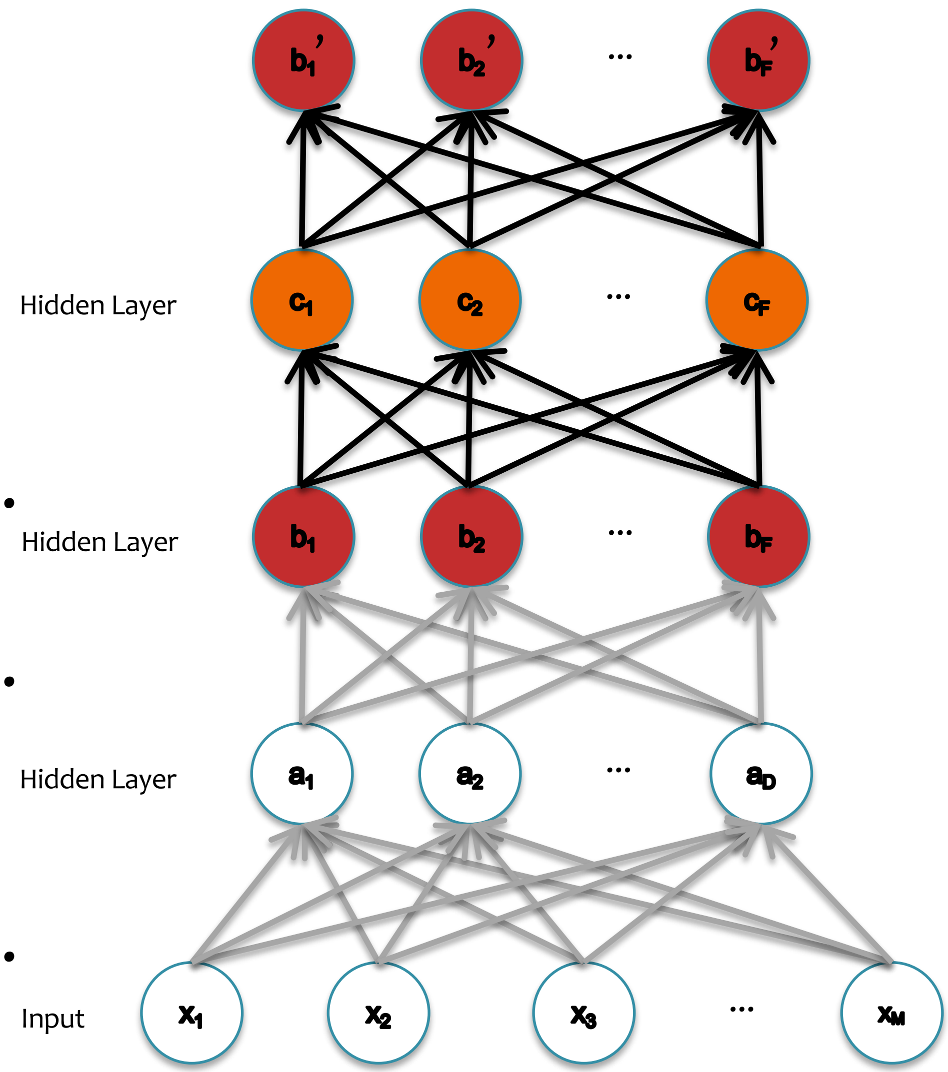
- Work bottom-up
  - Train hidden layer 1. Then fix its parameters.
  - Train hidden layer 2. Then fix its parameters.
  - …
  - Train hidden layer n. Then fix its parameters.



Unsupervised Pre-training

Unsupervised pre-training

- Work bottom-up
  - Train hidden layer 1. Then fix its parameters.
  - Train hidden layer 2. Then fix its parameters.
  - …
  - Train hidden layer n. Then fix its parameters.

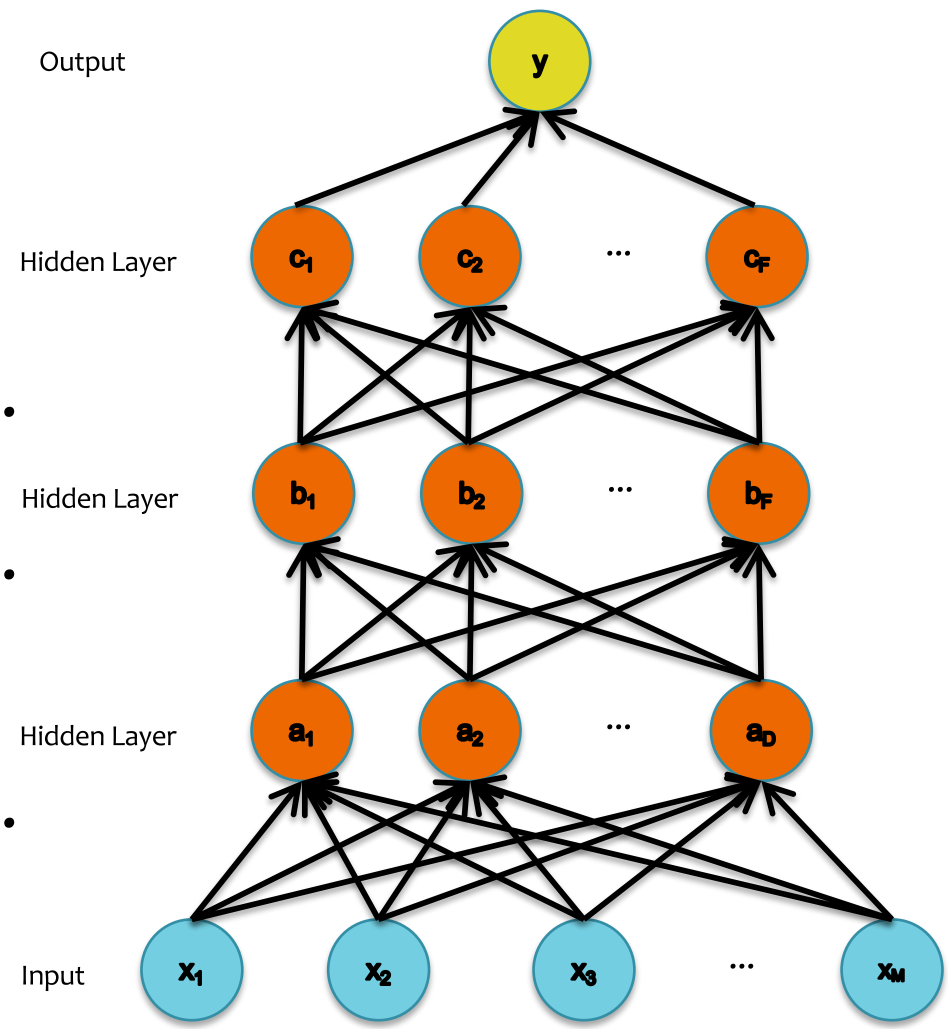


Unsupervised Pre-training

Unsupervised pre-training

- Work bottom-up
  - Train hidden layer 1. Then fix its parameters.
  - Train hidden layer 2. Then fix its parameters.
  - …
  - Train hidden layer n. Then fix its parameters.

Supervised fine-tuning
Backprop and update all parameters



Deep Network Training

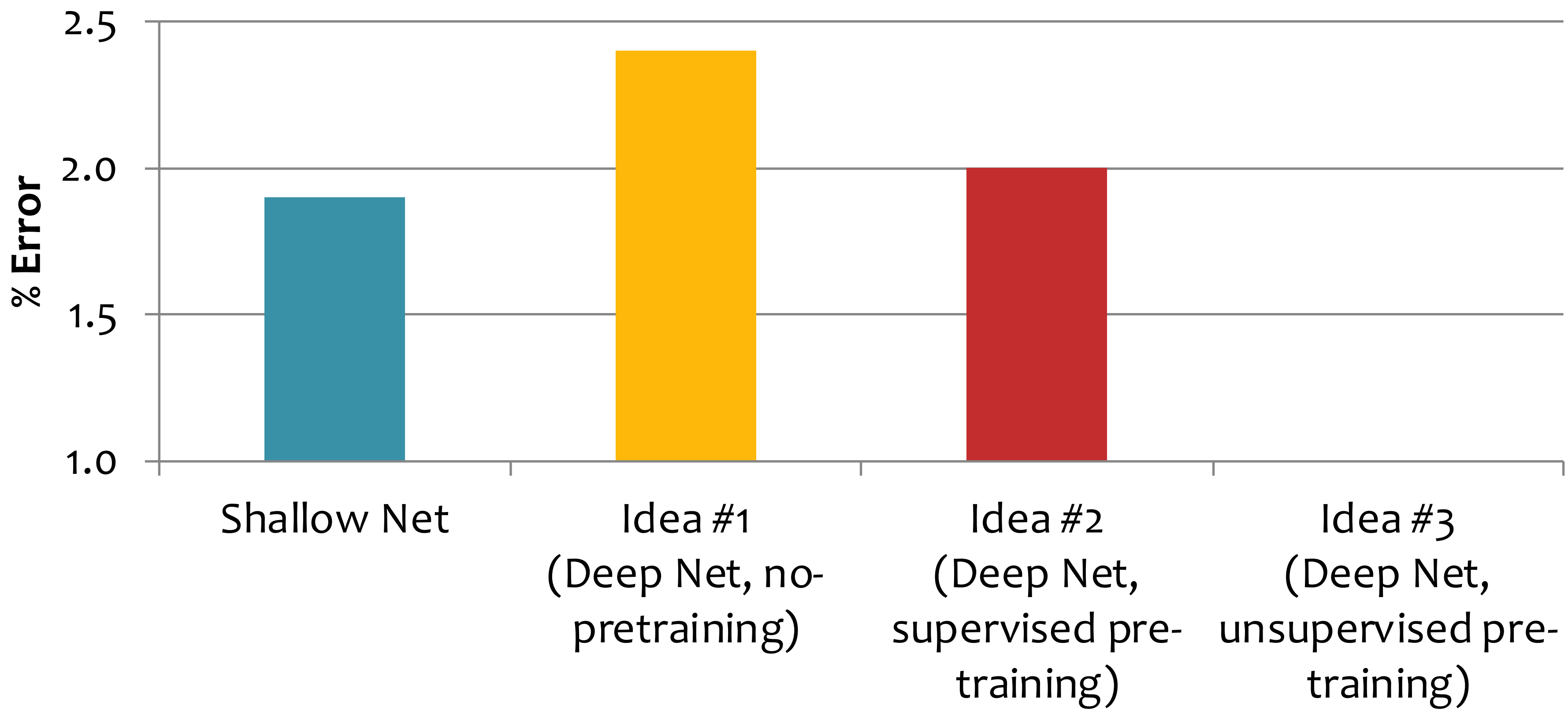
- **Idea #1:**
  1. Supervised fine-tuning only

- **Idea #2:**
  1. Supervised layer-wise pre-training
  2. Supervised fine-tuning

- **Idea #3:**
  1. Unsupervised layer-wise pre-training
  2. Supervised fine-tuning

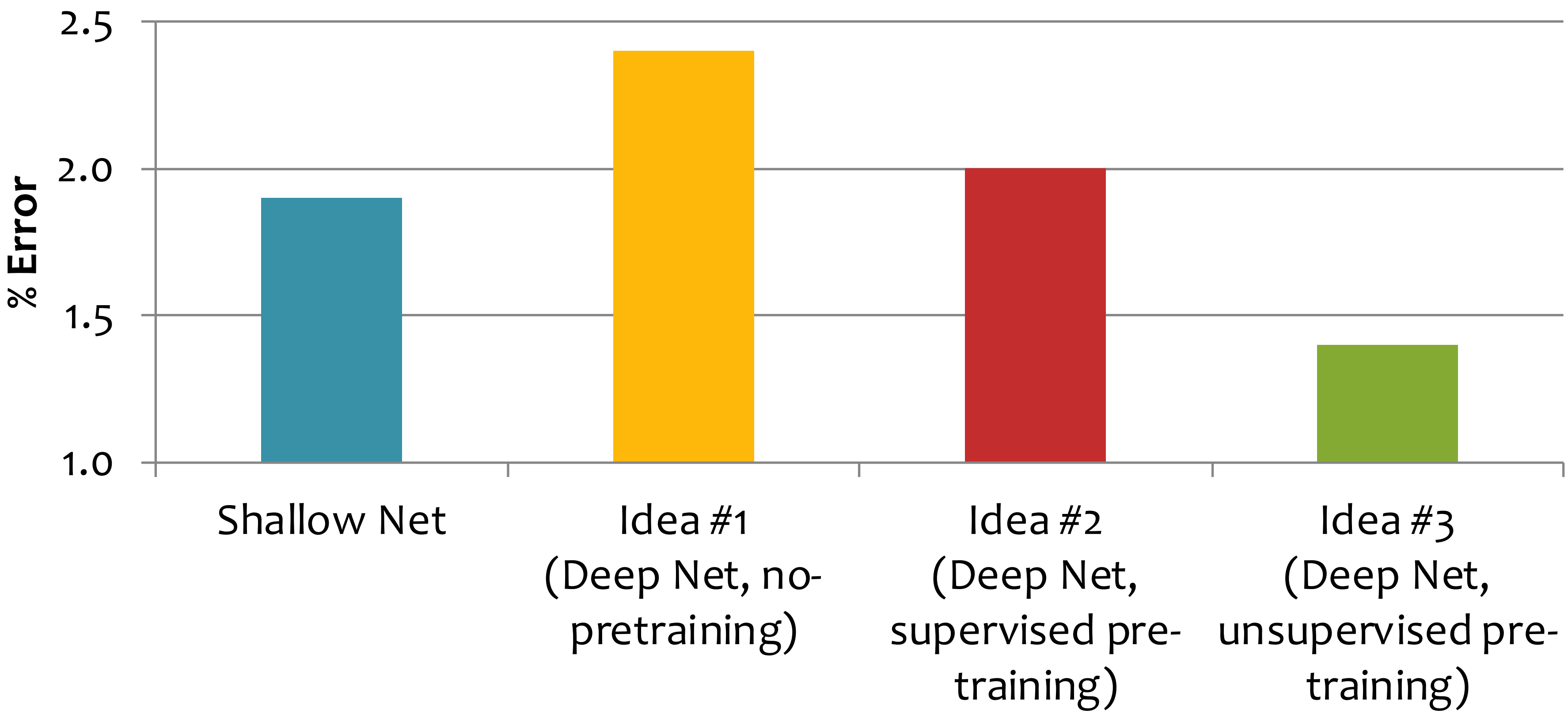
Comparison on MNIST

- Results from Bengio et al. (2006) on MNIST digit classification task
- Percent error (lower is better)



Comparison on MNIST

- Results from Bengio et al. (2006) on MNIST digit classification task
- Percent error (lower is better)



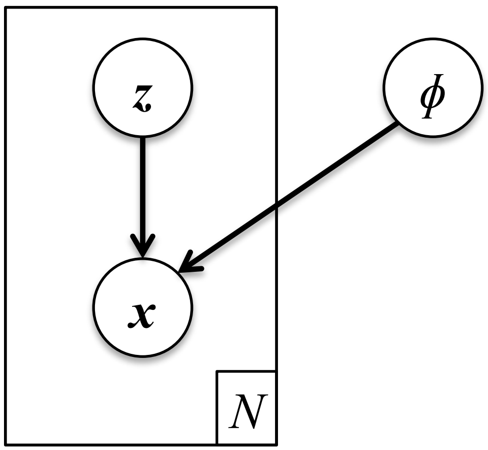
VARIATIONAL AUTOENCODERS

Why VAEs?

- **Autoencoders:**
  - learn a low dimensional representation of the input, but hard to work with as a generative model
  - one of the key limitations of autoencoders is that we have no way of **sampling** from them!
- **Variational autoencoders (VAEs)**
  - by contrast learn a continuous latent space that is **easy to sample from**!
  - can **generate** new data (e.g. images) by sampling from the learned generative model

Variational Autoencoders

$p_{\phi}(\mathbf{x}, \mathbf{z})$



$z \sim \text{Gaussian}(0, I)$

Graphical Model Perspective

- The DGM diagram shows that the VAE model is quite simple as a graphical model (ignoring the neural net details that give rise to **x**)
- Sampling from the model is easy:
  - Consider a DGM where $\mathbf{x} = g_{\phi}(\mathbf{z}/10 + \mathbf{z}/||\mathbf{z}||)$ (i.e. we don't use parameters ϕ)
  - Then we can draw samples of **z** and directly convert them to values **x**
- **Key idea of VAE:** define $g_{\phi}(z)$ as a neural net and learn ϕ from data

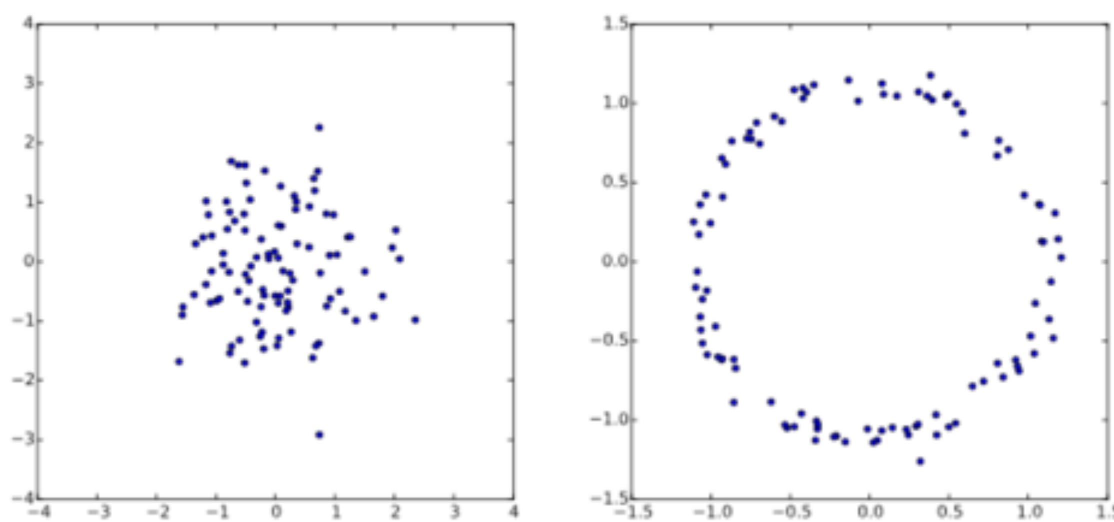


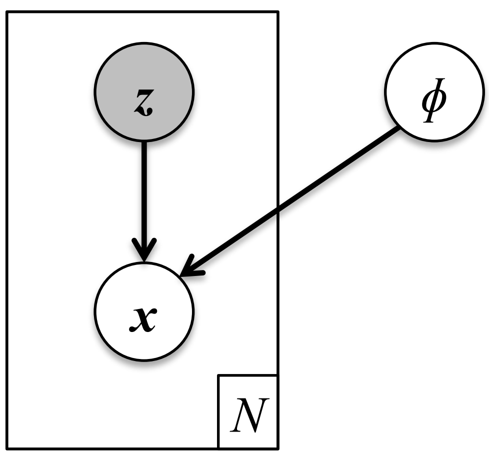
Figure from Doersch (2016)

Variational Autoencoders

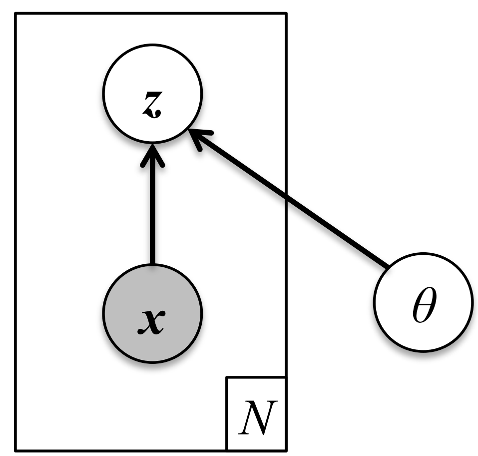
Neural Network Perspective

- We can view a variational autoencoder (VAE) as an autoencoder consisting of two neural networks
- VAEs (as encoders) define two distributions:
  - **encoder**: $q_\theta(\mathbf{z} \mid \mathbf{x})$
  - **decoder**: $p_\phi(\mathbf{x} \mid \mathbf{z})$
- Parameters θ and ϕ are neural network parameters (i.e. θ are not the variational parameters)

$$p_\phi(\mathbf{x} \mid \mathbf{z})$$



$$q_\theta(\mathbf{z} \mid \mathbf{x})$$

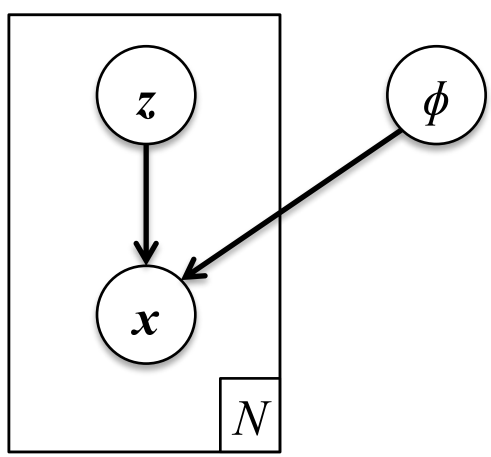


Variational Autoencoders

Graphical Model Perspective

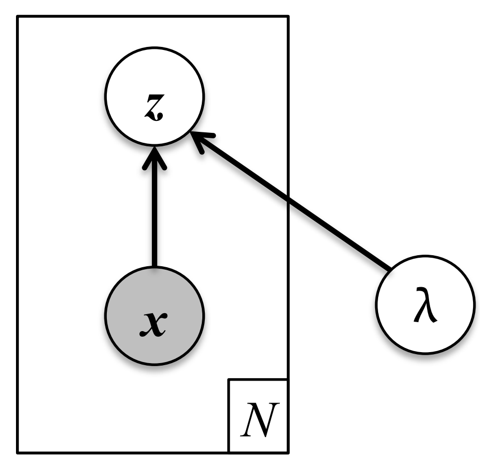
- We can also view the VAE from the perspective of variational inference
- In this case we have two distributions:
  - **model**: $p_\phi(\mathbf{z} \mid \mathbf{x})$
  - **variational approximation:** $q_{\lambda=f(x;\,\theta)}(\mathbf{z} \mid \mathbf{x})$
- We have the same model parameters ϕ
- The variational parameters λ are a function of NN parameters θ

$$p_\phi(\mathbf{x}, \mathbf{z})$$



$$z \sim \text{Gaussian}(0, I)$$

$$q_\lambda(\mathbf{z} \mid \mathbf{x})$$



$$\lambda = f(x;\, \theta)$$

Variational Autoencoders

Whiteboard

- Variational Autoencoder = VAE
- VAE as a Probability Model
- Parameterizing the VAE with Neural Nets
- Variational EM for VAEs

Reparameterization Trick

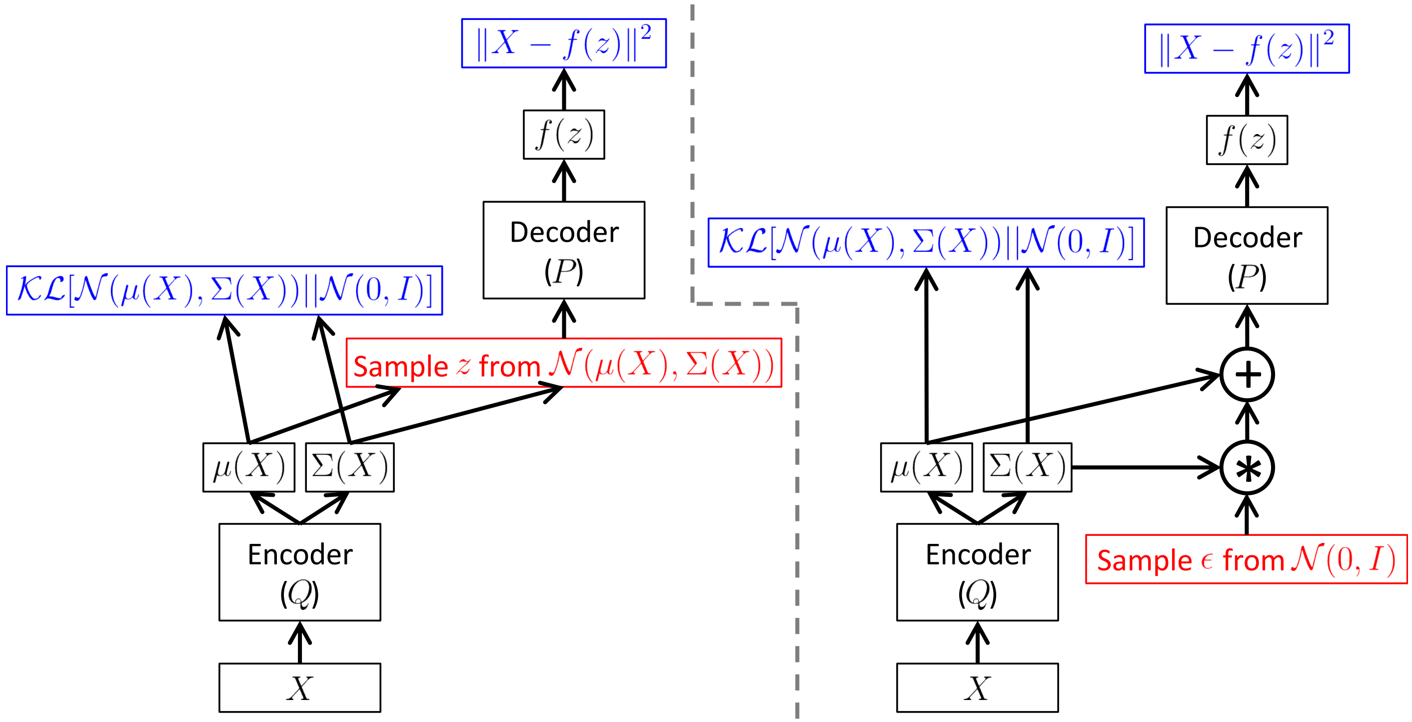


Figure 4: A training-time variational autoencoder implemented as a feed-forward neural network, where $P(X|z)$ is Gaussian. Left is without the "reparameterization trick", and right is with it. Red shows sampling operations that are non-differentiable. Blue shows loss layers. The feedforward behavior of these networks is identical, but backpropagation can be applied only to the right network.

Figure from Doersch (2016)