

Course number: 80240743

# Deep Learning

Xiaolin Hu (胡晓林) & Jun Zhu (朱军)

Dept. of Computer Science and Technology

Tsinghua University

# Deep Generative Models

## Part 2: Variational Auto-Encoders

**Jun Zhu**

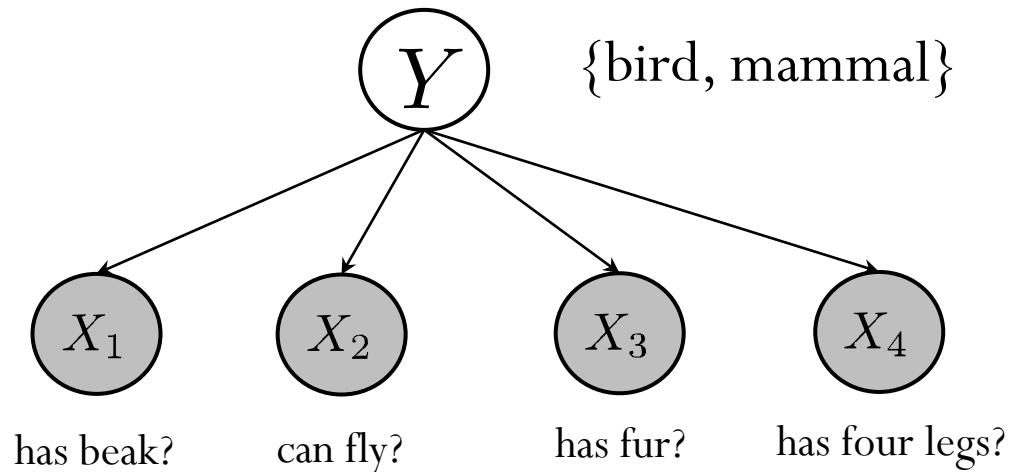
`dczsj@mail.tsinghua.edu.cn`

Department of Computer Science and Technology

Tsinghua University

# Recap of Basics of Generative Models

## ◆ Naïve Bayes classifier



A joint distribution:

$$p(\mathbf{x}, y) = \overset{\text{prior}}{p(y)} \overset{\text{likelihood}}{p(\mathbf{x}|y)}$$

Bayes' decision rule:

$$y^* = \arg \max_{y \in \mathcal{Y}} p(y|\mathbf{x})$$

# Recap of Basics of Generative Models

## ◆ Mixture of Gaussians

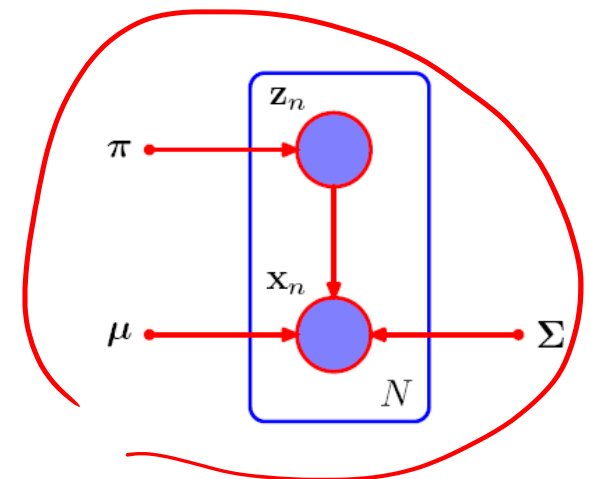
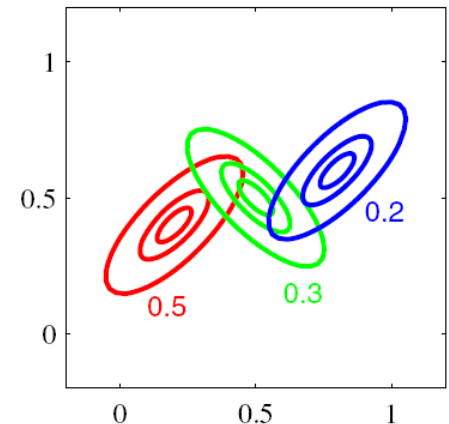
$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

## ◆ Equivalent representation:

$$\mathbf{z} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

$$p(\mathbf{x}, \mathbf{z}) = \prod_{k=1}^K \pi_k^{z_k} \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)^{z_k}$$

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z})$$



# Recap of Basics of Generative Models

- ◆ Maximize the lower bound or minimize the gap:

$$\log p(\mathcal{D}|\Theta) \geq \sum_{n=1}^N \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log \left( \frac{p(\mathbf{x}_n, \mathbf{z}_n)}{q(\mathbf{z}_n)} \right) \triangleq \mathcal{L}(\Theta, q(\mathbf{Z}))$$

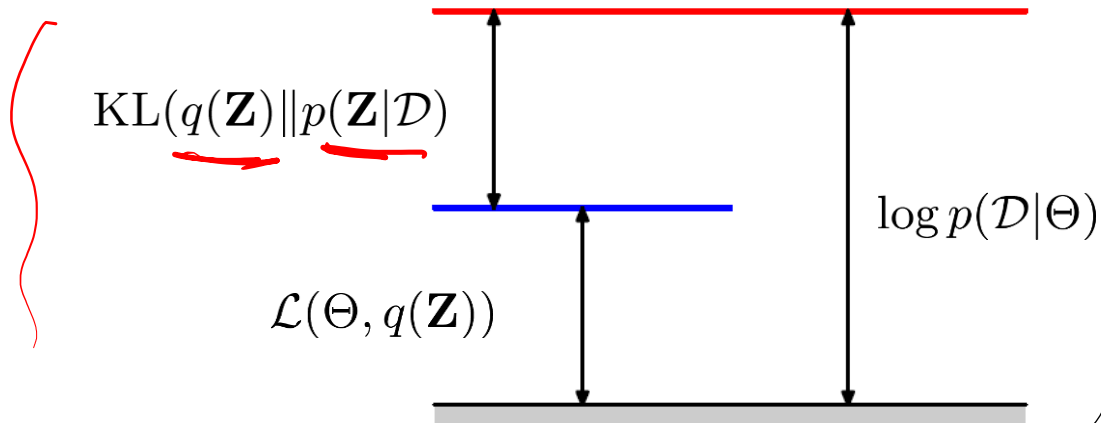
- Maximize over  $q(\mathbf{Z}) \Rightarrow$  E-step

- Maximize over  $\Theta \Rightarrow$  M-step

$$\text{KL}(q(\mathbf{Z}) \| p(\mathbf{Z}|\mathcal{D}))$$

$$\mathcal{L}(\Theta, q(\mathbf{Z}))$$

$$\log p(\mathcal{D}|\Theta)$$



# Language model revisited

- ◆ A simple unigram language model
  - ▣ Observations (e.g., bag-of-words)

$$\mathbf{x} = \{x_1, \dots, x_d\}$$

---

## Racing Thompson: an Efficient Algorithm for Thompson Sampling with Non-conjugate Priors

---

Anonymous Author(s)  
Affiliation  
Address  
email

### Abstract

Thompson sampling has impressive empirical performance for many multi-armed bandit problems. But current algorithms for Thompson sampling only work for the case of conjugate priors since these algorithms require to infer the posterior, which is often computationally intractable when the prior is not conjugate. In this paper, we propose a novel algorithm for Thompson sampling which only requires to draw samples from a tractable distribution, so our algorithm is efficient even when the prior is non-conjugate. To do this, we reformulate Thompson sampling as an optimization problem via the Gumbel-Max trick. After that we construct a set of random variables and our goal is to identify the one with highest mean. Finally, we solve it with techniques in best arm identification.

### 1 Introduction

In multi-armed bandit (MAB) problems [20], an agent chooses an action (in the literature of MAB, an action is also named as an arm.) from an action set repeatedly, and the environment returns a reward as a response to the chosen action. The agent's goal is to maximize the cumulative reward over a period of time. In MAB, a reward distribution is associated with each arm to characterize the uncertainty of the reward. One key issue for MAB and many on-line learning problems [3] is to well-balance the exploitation-exploration tradeoff, that is, the tradeoff between choosing the action that has already yielded greatest rewards and the action that is relatively unexplored.



Term	D1	D2
game	1	0
decision	0	0
theory	2	0
probability	0	3
analysis	0	2
...		

# Language model revisited

- ◆ A simple unigram language model

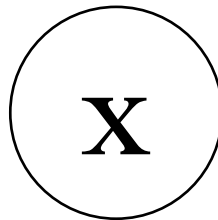
- ▣ Observations (e.g., bag-of-words)

$$\mathbf{x} = \{x_1, \dots, x_d\}$$

- ▣ Joint distribution (likelihood)

$$p(\mathbf{x}; \theta) = \prod p(x_i ; \theta)$$

- ▣ Graphical representation (parameters ignored)



# Language model revisited

- ◆ A fully-observed model is not sufficient
- Data has hidden structures

## Generalized BROOF-L2R: A General Framework for Learning to Rank Based on Boosting and Random Forests

Clebson C. A. de Sá Marcos A. Gonçalves Daniel X. Sousa Thiago Salles  
Federal University of Minas Gerais  
Computer Science Department  
Belo Horizonte, Brazil  
{clebsonc, mgoncalv, danieleks, salles}@dcc.ufmg.br

### ABSTRACT

The task of retrieving information that really matters to the users is considered hard when taking into consideration the extent and increasing amount of available information. To improve the effectiveness of the information seeking task, systems have tried to be the combination of many procedures for several machine learning techniques. A task often known as learning to rank (LTR). The most effective learning methods for this task are based on ensemble of trees (e.g., Random Forests) and/or boosting techniques (e.g., Boosting, XGBoost, LightGBM). In this paper, we propose a general framework that smoothly combines ensemble of additive trees, specifically Boosting forests, with Boosting in a regular way for the task of LTR. In particular, we exploit out-of-bag sampling as well as a selective weight updating strategy (inspired by the out-of-bag sampling) to effectively reduce the training performance. We illustrate such a general framework by considering different loss functions, different ways of regularizing the ensemble, and different ways of regularizing the ensemble.

## Document Retrieval Using Entity-Based Language Models

Keywords  
Learning to Rank

Hadas Raviv  
Technion, Israel  
hadasrv@tx.technion.ac.il

Oren Kurland  
Technion, Israel  
kurland@ie.technion.ac.il

David Carmel  
Yahoo Research, Israel  
david.carmel@gmail.com

### ABSTRACT

We address the ad hoc document retrieval task by devising novel types of entity-based language models. The models utilize information about single terms in the query and documents as well as term co-occurrences related to those (marked) in the text and which often use auxiliary information about entities from the entity repository (e.g., textual description of entities, entities' categories and inter-entity relationships). We show that the models significantly outperform the state-of-the-art in the entity-based retrieval task. Empirical evaluation demonstrates the merits of using the language models for retrieval. For example, the performance translates that of a state-of-the-art term proximity method. We also show that the language models can be effectively used for cluster-based document retrieval and query expansion.

Categories and Subject Descriptors: H.3.3 Information Search and Retrieval; Retrieval models

Keywords: document retrieval; entity-based language models

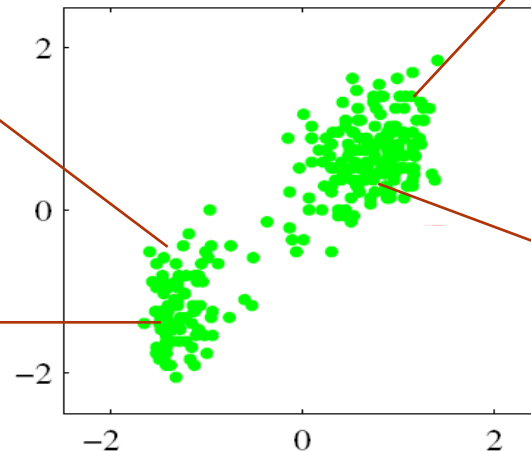
### 1. INTRODUCTION

Most ad hoc document retrieval methods compare query and document representations. To address the potential vocabulary mismatch between a short query and documents relevant to the query, various semantic-document-query similarity measures have been proposed [26]. Specifically, there is a growing body of work on retrieval

in the text and wordings of entities in it, along with two corpus-based relevance statistics. This is in contrast to expansion-based and projection-based representations that utilize also terms and entities related to those (marked) in the text and which often use auxiliary information about entities from the entity repository (e.g., textual description of entities, entities' categories and inter-entity relationships). We show that the models significantly outperform the state-of-the-art in the entity-based retrieval task. Empirical evaluation demonstrates the merits of using the language models for retrieval. For example, the performance translates that of a state-of-the-art term proximity method. We also show that the language models can be effectively used for cluster-based document retrieval and query expansion.

The reason for addressing the question just posed is twofold. First, it will shed light on the effectiveness of using entities in their most basic capacity, that is, spatial features marked by queries and documents. Indeed, findings in past work on ad hoc retrieval regarding the merits of using surface-level entity-based representations are inconclusive [16, 42, 47, 3, 14]. Second, such representations can be naturally used in creating retrieval expansion and leads to improve performance, e.g., query expansion and cluster-based document retrieval as we show in this paper.

There are various potential merits in using surface-level entity-based representations. For example, there can help to cope with the vocabulary mismatch problem, e.g., the entity (United States of America) can have different expressions in the text, including, "U.S.", "USA", "United States" and



## Stochastically Transitive Models for Pairwise Comparisons: Statistical and Computational Issues

Nihar B. Shah<sup>1</sup>

Souravranjan Halderdhar<sup>2</sup>

Adityanand Ganeshaiah<sup>2</sup>

Martin J. Wainwright<sup>1</sup>

<sup>1</sup>Dept. of EECS, Univ. of California, Berkeley

<sup>2</sup>Dept. of Statistics, Carnegie Mellon University

NIBAH@EECS.BERKELEY.EDU

SIVA@STAT.CMU.EDU

ADITYA@STAT.BERKELEY.EDU

WAINW@STAT.BERKELEY.EDU

### Abstract

There are various parametric models for analyzing pairwise comparison data, including the Bradley-Terry-Luce (BTL) and Thurstone models, but their reliance on strong parametric assumptions is limiting. In this work, we study a flexible model for pairwise comparisons, under which the probabilities of outcomes are required only to satisfy a natural form of stochastic transitivity. This class includes parametric models including the BTL and Thurstone models in special cases, but is considerably more general. We provide various examples of models in this broader stochastically transitive class for which classical parametric models provide poor fits. Despite this greater flexibility, we show that the matrix of probabilities can be estimated at the same rate as in standard parametric model-based, unlike in the BTL and T competing the maximum-likelihood

ing, online search rankings, and ad placement problems. In rough terms, given a set of  $n$  objects, and a collection of possibly inconsistent comparisons between pairs of these objects, the goal is to aggregate these comparisons in order to perform effective statistical inference on various underlying properties of the population. A particular property of interest is the underlying pairwise comparison probabilities—that is, the probability that object  $i$  is preferred to object  $j$  in a pairwise comparison. The Bradley-Terry-Luce (Bradley & Terry, 1952; Luce, 1959) and Thurstone (Thurstone, 1927) models are natural ways of analyzing this type of pairwise comparison data. These models are parametric in nature: more specifically, they assume the existence of an  $n$ -dimensional weight vector that measures the quality or strength of each item. The pairwise comparison probabilities are then determined via some fixed (parametric) function of the qualities of the pair of objects.

## No Oups, You Won't Do It Again: Mechanisms for Self-correction in Crowdsourcing

Nihar B. Shah

Dept. of EECS, University of California, Berkeley

Dengyong Zhou

Microsoft Research, Redmond

NIBAH@EECS.BERKELEY.EDU

DENGYONG.ZHOU@MICROSOFT.COM

### Abstract

Crowdsourcing is a very popular means of obtaining the large amounts of labeled data that modern machine learning methods require. Although cheap and fast to obtain, crowdsourced labels suffer from significant amounts of error, thereby degrading the performance of downstream machine learning tasks. With the goal of improving the quality of the labeled data, we seek to mitigate the many errors that occur due to systematic or inadvertent errors by crowdsourcing workers. We propose a two-stage setting for crowdsourcing where the worker first answers the questions, and is then allowed to change her answers after looking at a majority reference answer. We mathematically formalize this process and develop mechanisms to incentivize workers to act appropriately. Our mathematical guarantees show that our mechanisms incentivize the workers to answer honestly in both

the Internet typically in exchange from some monetary payments. Crowdsourcing is widely used in many real-world applications, and is particularly popular for collecting training labels for machine learning powered systems like web search engines (Berges et al., 2005; Alonso & Metzger, 2009; Kazai, 2011) or to supplement automated algorithms (Khuri et al., 2011; Ling & Rao, 2011; Yao et al., 2008). The labels obtained from crowdsourcing, however, have significant amounts of error (Kazai et al., 2011; Vassent et al., 2011; Wain et al., 2010), thereby degrading the performance of the machine learning algorithms that use this data downstream. Consequently, there is much emphasis on gathering higher quality labels, since a lower noise implies requirement of fewer labels for obtaining the same accuracy in practice.

In a study from a few years back, Kalanman & Frederick (2002) asked the following question to many participants: "A bat and ball cost a dollar and ten cents. The bat costs a dollar more than the ball. How much does the ball cost?" (See also The New Yorker (2012).) A large number of re-

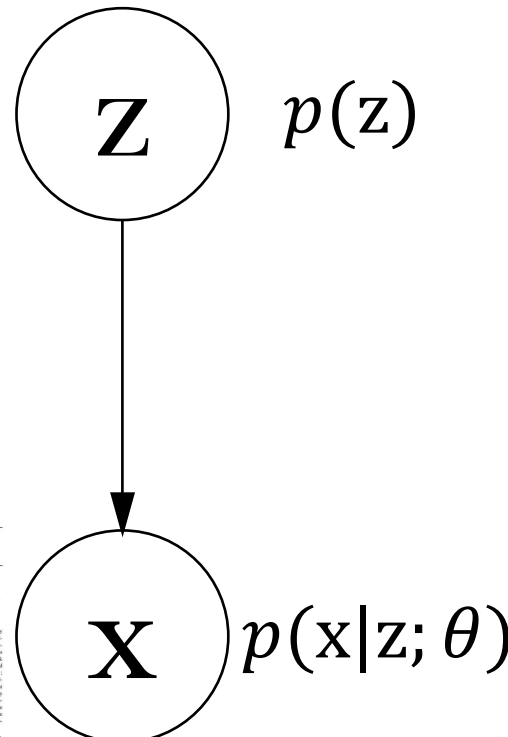
- A simple distribution is not sufficient





# Language model revisited

- ◆ Mixture model --- a simple generative model with hidden factors
- Graphical model representation



$$p(x, z) = p(z)p(x|z; \theta)$$

## Generalized BROOF-L2R: A General Framework for Learning to Rank Based on Boosting and Random Forests

Cleison C. A. de Sá, Marcos A. Gonçalves, Daniel X. Sousa, Thiago Sales  
Federal University of Bahia Campus  
Computer Science Department  
Belo Horizonte, Brazil  
{cleisonc, mgoncalv, dsousa, tsales}@dcc.ufba.br

**ABSTRACT**  
This task of retrieving information that really matters to the users is considered hard when taking into consideration the relevant and irrelevant information. In response to this task, various have called attention to various of such

learning to rank methods as being suitable for this task, and have proposed such methods were able to compare considered separately, only original training set and

**Keywords**  
Learning to Rank, L2R

**1. INTRODUCTION**  
There are various parametric models being proposed, such as Boosting, Linear SVM (L2R) and others, for their ability to learn from complex data. In this work, we propose a new parametric model for learning to rank, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data.

**2. RELATED WORK**  
There are various parametric models being proposed, such as Boosting, Linear SVM (L2R) and others, for their ability to learn from complex data. In this work, we propose a new parametric model for learning to rank, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data.

**3. PROPOSED METHOD**  
This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data.

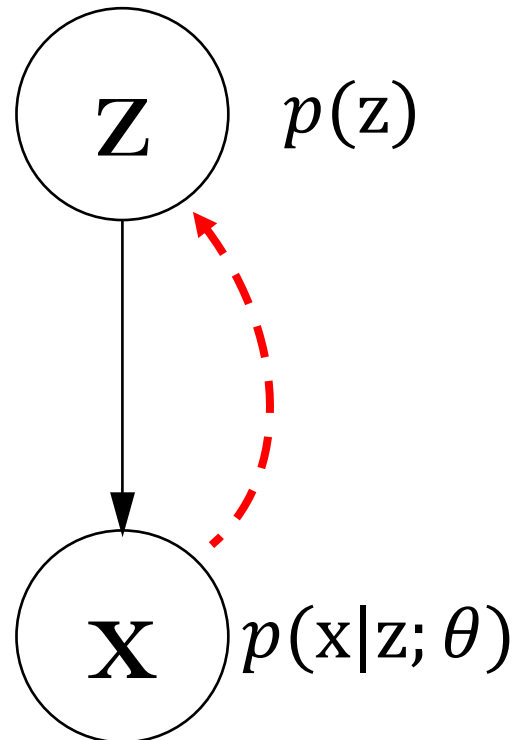
**4. EXPERIMENTAL EVALUATION**  
This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data.

**5. CONCLUSION**  
This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data.

**6. REFERENCES**  
This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data. This model is able to learn from complex data, which is able to learn from complex data.

# Language model revisited

- ◆ Mixture model --- a simple generative model with hidden factors
  - Infer the latent  $Z$ :



Bayes' Rule:

$$p(z|x) = \frac{p(x, z)}{p(x)} \\ \propto p(z)p(x|z; \theta)$$

No Oops, You Won't Do It Again:  
Mechanisms for Self-correction in Crowdsourcing

Nihar B. Shah  
Dept. of EECS, University of California, Berkeley  
Dengyong Zhou  
Microsoft Research, Redmond

NIHAR@EECS.BERKELEY.EDU  
DENGYONG.ZHOU@MICROSOFT.COM

## Abstract

Crowdsourcing is a very popular means of obtaining the large amounts of labeled data that modern machine learning methods require. Although cheap and fast to obtain, crowdsourced labels suffer from significant amounts of error, thereby degrading the performance of downstream machine learning tasks. With the goal of improving the quality of the labeled data, we seek to mitigate the many errors that occur due to silly mistakes or inadvertent errors by crowdsourcing workers. We propose a two-stage setting for crowdsourcing where the worker first answers the questions, and is then allowed to change her answers after looking at a (noisy) reference answer. We mathematically formalize this process and develop mechanisms to incentivize workers to act appropriately. Our mathematical guarantees show that our mechanism incentivizes the workers to answer honestly in both

the Internet typically in exchange for some monetary payments. Crowdsourcing is widely used in many real-world applications, and is particularly popular for collecting training labels for machine learning powered systems like web search engines (Burgess et al., 2005; Alonso & Mizzaro, 2009; Kazai, 2011) or to supplement automated algorithms (Kash et al., 2011; Lang & Rio-Rousse, 2011; Van Ahn et al., 2008). The labels obtained from crowdsourcing, however, have significant amounts of error (Kazai et al., 2011; Vassiri et al., 2011; Wals et al., 2010), thereby degrading the performance of the machine learning algorithms that use this data downstream. Consequently, there is much emphasis on gathering higher quality labels, since a lower noise implies requirement of fewer labels for obtaining the same accuracy in practice.

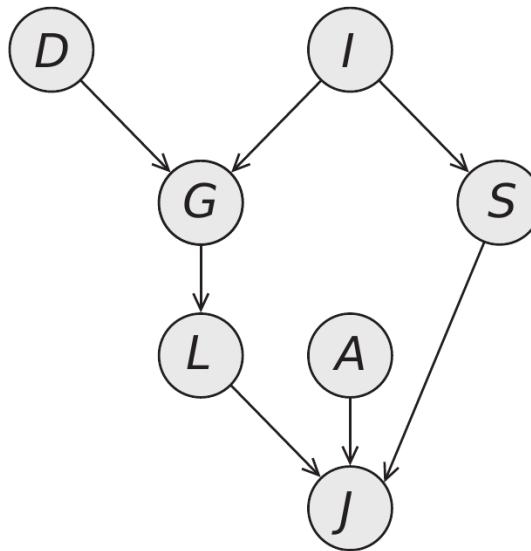
In a study from a few years back, Kahneman & Frederick (2002) asked the following question to many participants: "A bat and ball cost a dollar and ten cents. The bat costs a dollar more than the ball. How much does the ball cost?" (See also The New Yorker (2012).) A large number of re-



Going Deep ...

# Hierarchical Bayesian Modeling

- ◆ Build a hierarchy through distributions in analytical forms



$$P(D, I, G, S, L, A, J) = P(D)P(I)P(G|D, I)P(S|I) \cdot \\ P(A) P(L|G)P(J|L, A, S)$$

**Simple, Local Factors:** a conditional probability distribution

# Multiple Topics exist in a Document

◆ Go deeper into a single document

## Seeking Life's Bare (Genetic) Necessities

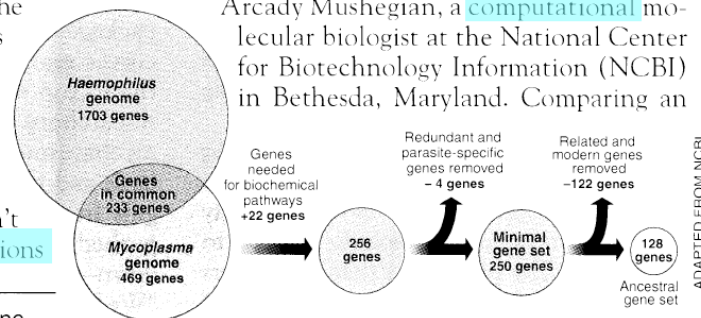
COLD SPRING HARBOR, NEW YORK—How many **genes** does an **organism** need to **survive**? Last week at the genome meeting here,\* two genome researchers with radically different approaches presented complementary views of the basic genes needed for **life**. One research team, using **computer** analyses to compare known **genomes**, concluded that today's **organisms** can be sustained with just 250 genes, and that the earliest life forms required a mere 128 **genes**. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those **predictions**

\* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

“are not all that far apart,” especially in comparison to the 75,000 **genes** in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a **genetic numbers** game, particularly as more and more **genomes** are completely mapped and sequenced. “It may be a way of organizing any newly **sequenced genome**,” explains Arcady Mushegian, a **computational** molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an



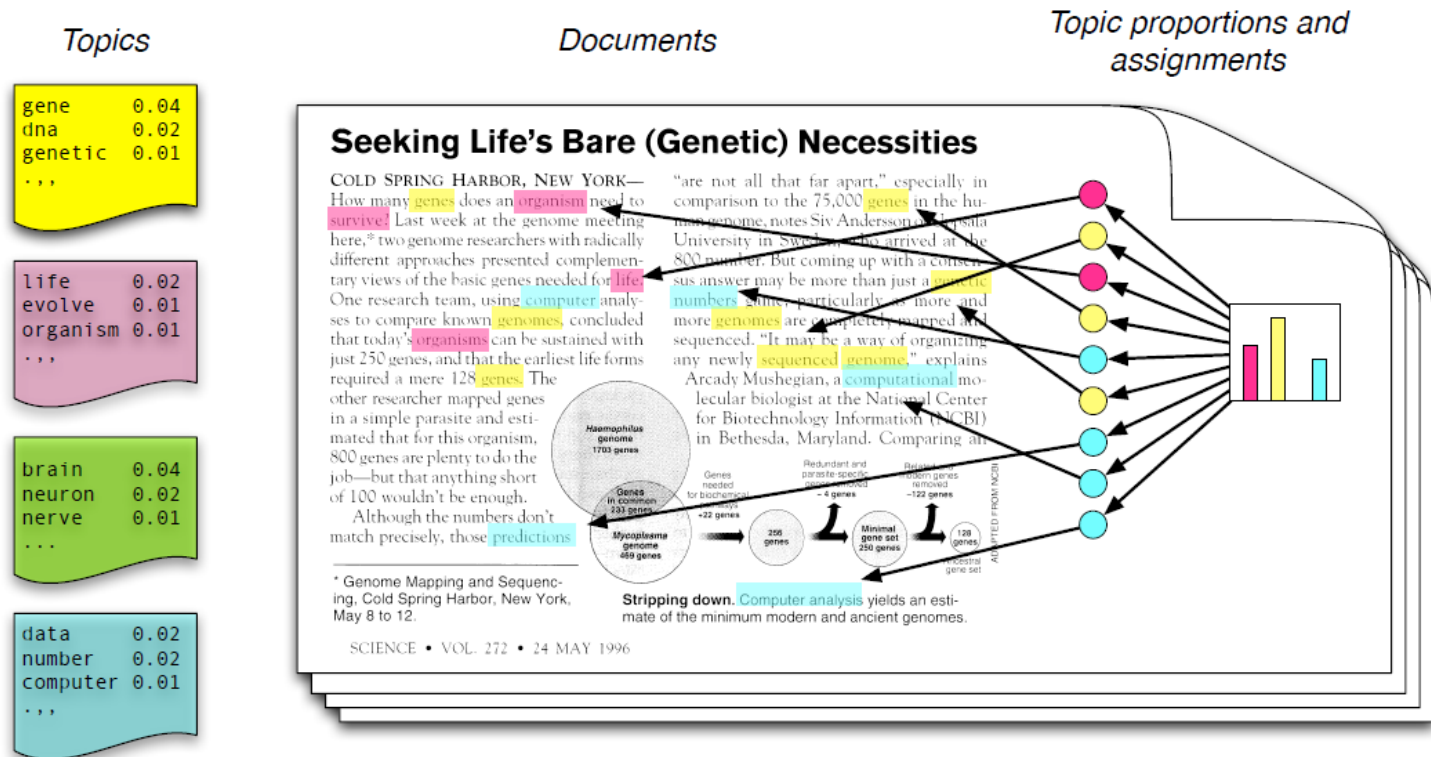
**Stripping down.** **Computer analysis** yields an estimate of the minimum modern and ancient genomes.

ADAPTED FROM NCBI

**Simple intuition:** Documents exhibit multiple topics.

[Slides courtesy: D. Blei]

# Latent Dirichlet Allocation (LDA)

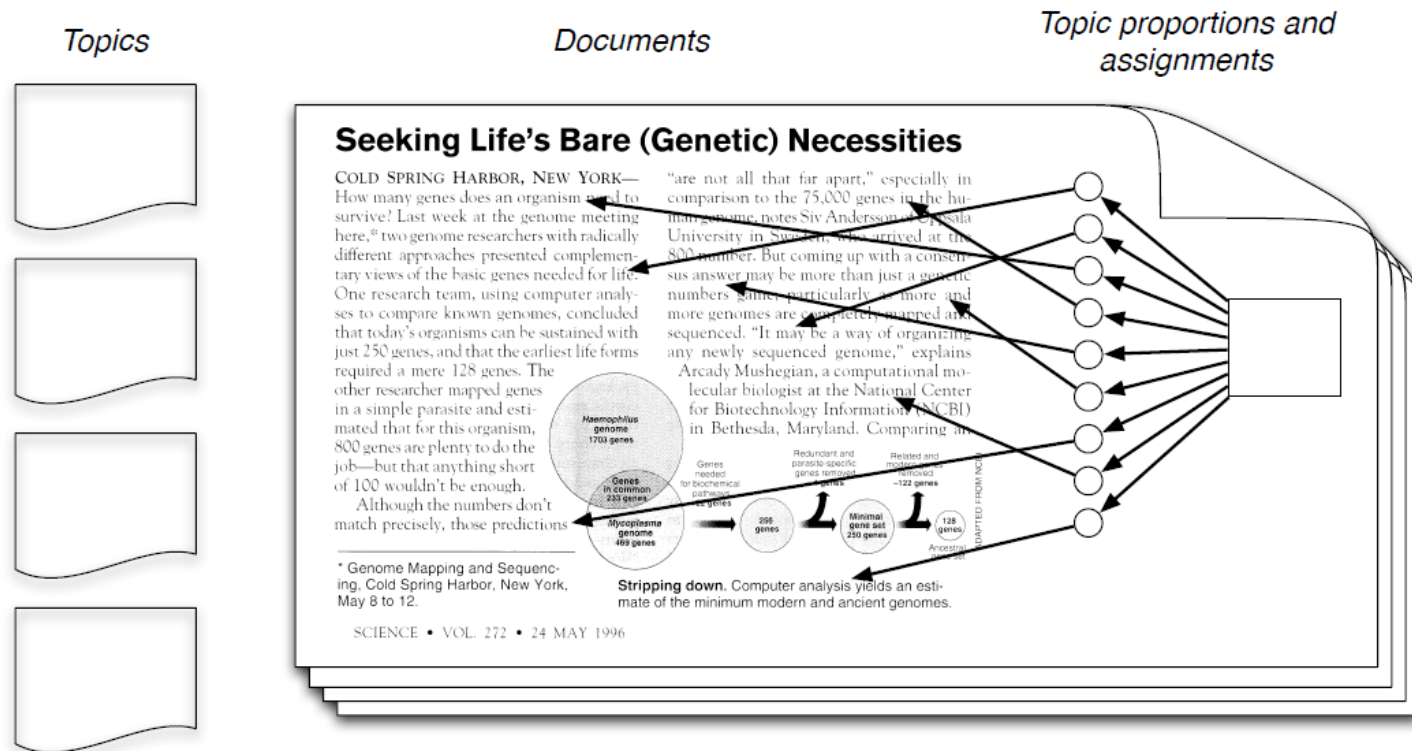


- Each **topic** is a distribution over words
- Each **document** is a mixture of corpus-wide topics
- Each **word** is drawn from one of those topics

[Slides courtesy: D. Blei]

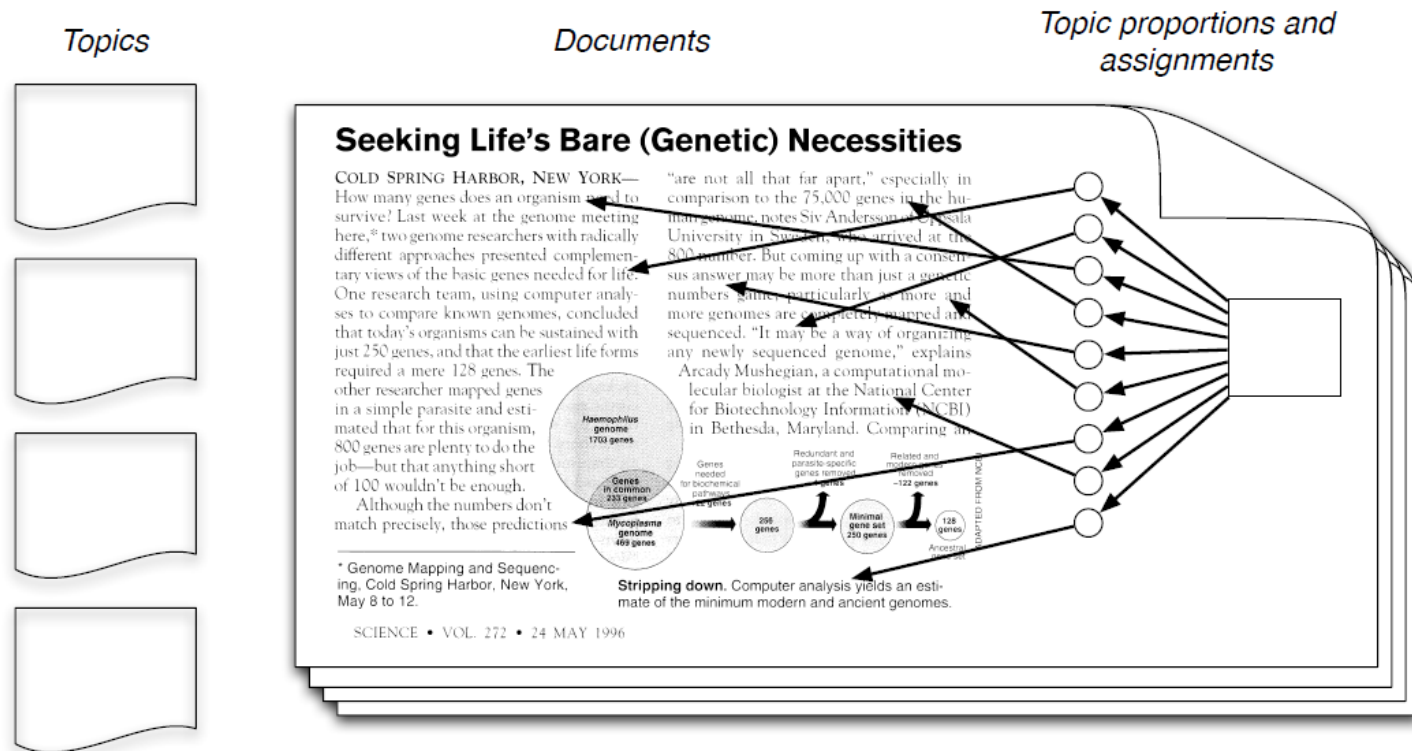


# Latent Dirichlet Allocation



- In reality, we only observe the documents
- The other structure are **hidden variables**

# Latent Dirichlet Allocation



- Our goal is to **infer** the hidden variables
- I.e., compute their distribution conditioned on the documents

$$p(\text{topics, proportions, assignments} | \text{documents})$$

[Slides courtesy: D. Blei]

## ◆ Applications

- Data visualization
- Natural language processing
- Recommendation system
- Computer vision
- Information retrieval
- and many more

## ◆ Alternative views

- Clustering
- Matrix factorization
- Discrete PCA

# Learning with Big Data

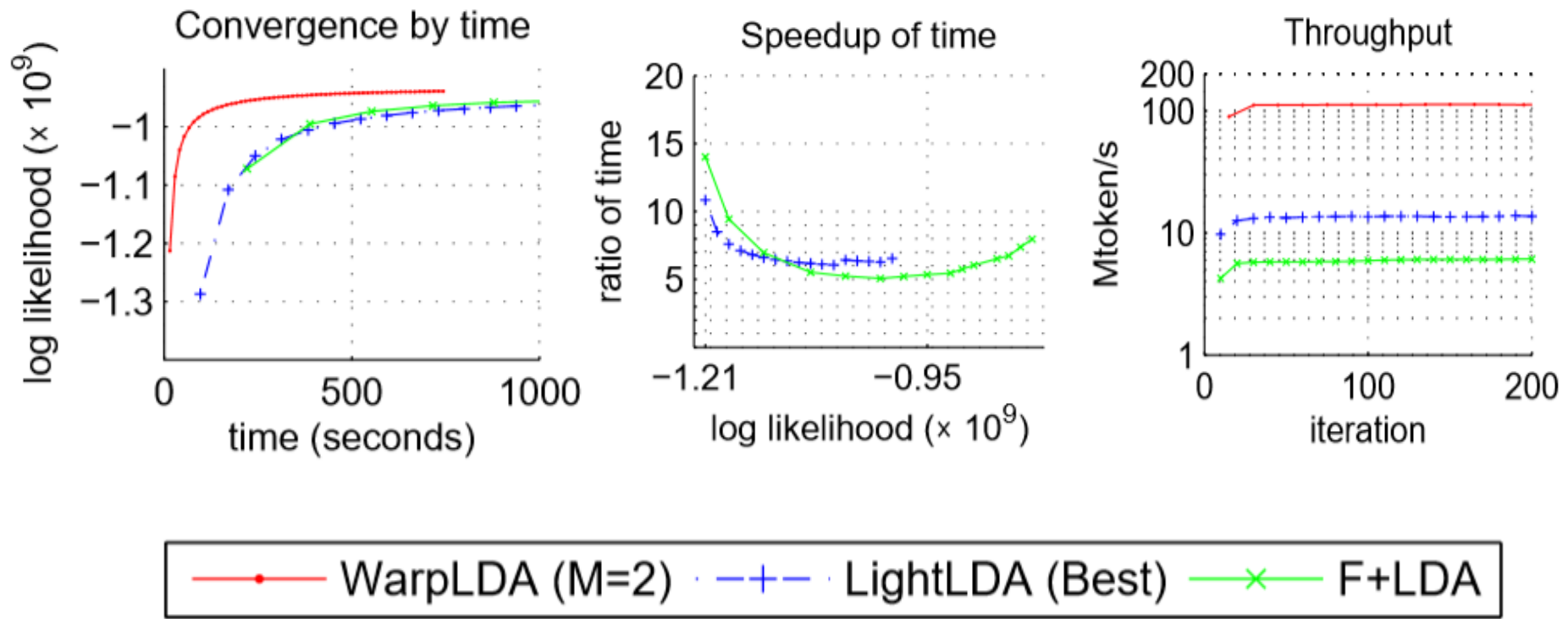
- ◆ Data can be very large
- ◆ The ClueWeb12 dataset (webpages)
  - #docs (D): billion
  - #vocabulary (V): million
  - #topics (K): million
  - #tokens (T): 100 billion
  - Raw size (html): 27TB
  - Text: 700GB
- ◆ Our recent work:
  - WarpLDA on CPU clusters (VLDB 2016)
  - SaberLDA on GPUs (ASPLOS 2017)
  - Scalable dynamic topic models (WWW 2016)
  - Online Bayesian passive-aggressive learning (JMLR 2017)
  - Scalable CTM (NIPS 2013)

# Learning with Big Data

- ◆ Data can be very large
- ◆ The ClueWeb12 dataset (webpages)
  - #docs (D): billion
  - #vocabulary (V): million
  - #topics (K): million
  - #tokens (T): 100 billion
  - Raw size (html): 27TB
  - Text: 700GB
- ◆ Our recent work:
  - WarpLDA on CPU clusters (VLDB 2016)
  - SaberLDA on GPUs (ASPLOS 2017)
  - Scalable dynamic topic models (WWW 2016)
  - Online Bayesian passive-aggressive learning (JMLR 2017)
  - Scalable CTM (NIPS 2013)

# WarpLDA

- ◆ Much faster convergence than previous methods



# WarpLDA

## ◆ Large data and model

### □ The ClueWeb12 dataset

- 600M documents

- 200G tokens

- 1M vocabulary

- 1M topics

- Cdk:  $600M * 1M$

- Cwk:  $1M * 1M$  sparse

- Ck: 1M

- Accessed sequentially

- Can be distributed

= 1600GB space to store

- Accessed randomly, but not in cache!

- Not even fit in single node

= 100GB-1TB space to store

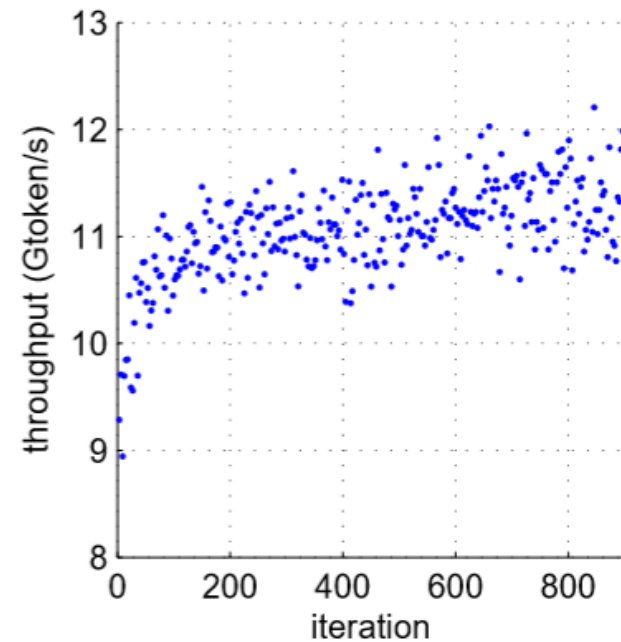
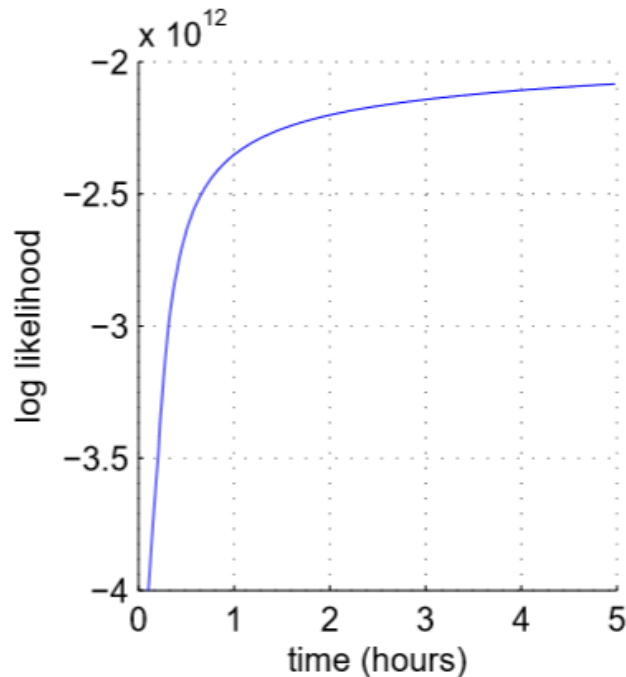
= 8MB to store

- Fits in L3 cache

# WarpLDA

## ◆ Very scalable

- 256 machines on Tianhe-2 achieves 12Gtoken/s
- Previous methods:  $\sim 100\text{Mtoken/s}$





# WarpLDA

◆ Mines meaningful topics from 600M web pages

机器学习: 26 topics

kernel machines learning workshop vector sch nips tutorials call smola  
workshop learning conference machine international systems intelligence artificial computational canada  
learning machine analysis data algorithms statistics computational theory complexity statistical

清华大学: 4 topics

iii iiis tsinghua titled yao itcs titles achievements introduction hrd  
university china tsinghua chinese beijing english peking wang shanghai zhang  
journal journals international applied tsinghua practice bulletin american issues annals  
ihp physics lnt beijing institute peking hep tsinghua energy cas

# Generative Topic Embedding

## ◆ Word Embedding

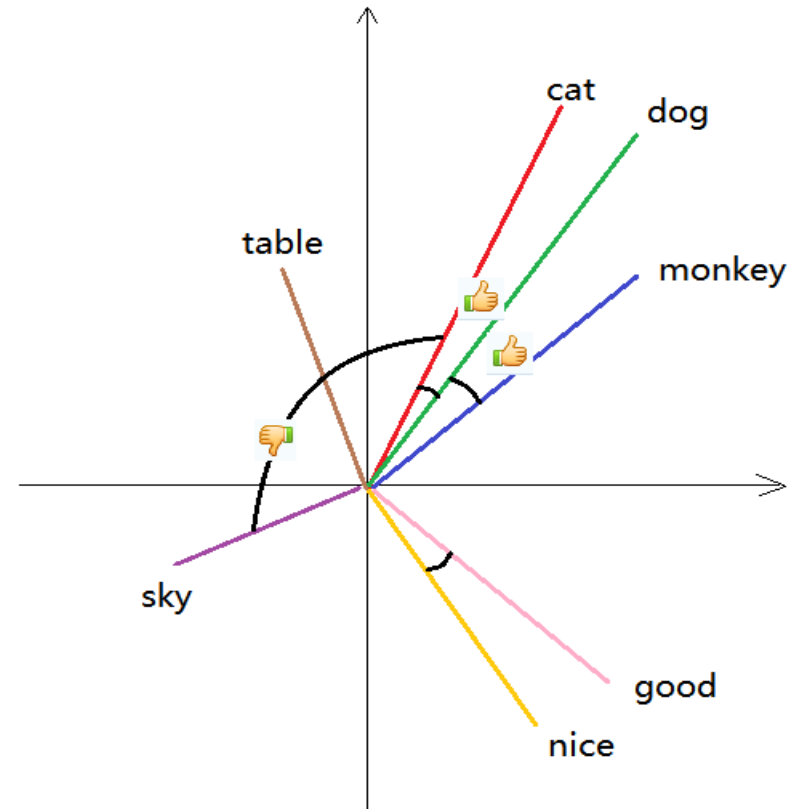
- Widely used in NLP applications
- Maps words into continuous, low dimensional vectors
- Vector direction usually encodes semantics

## ◆ Embedding methods

- Paragraph Vector, CNN, ...

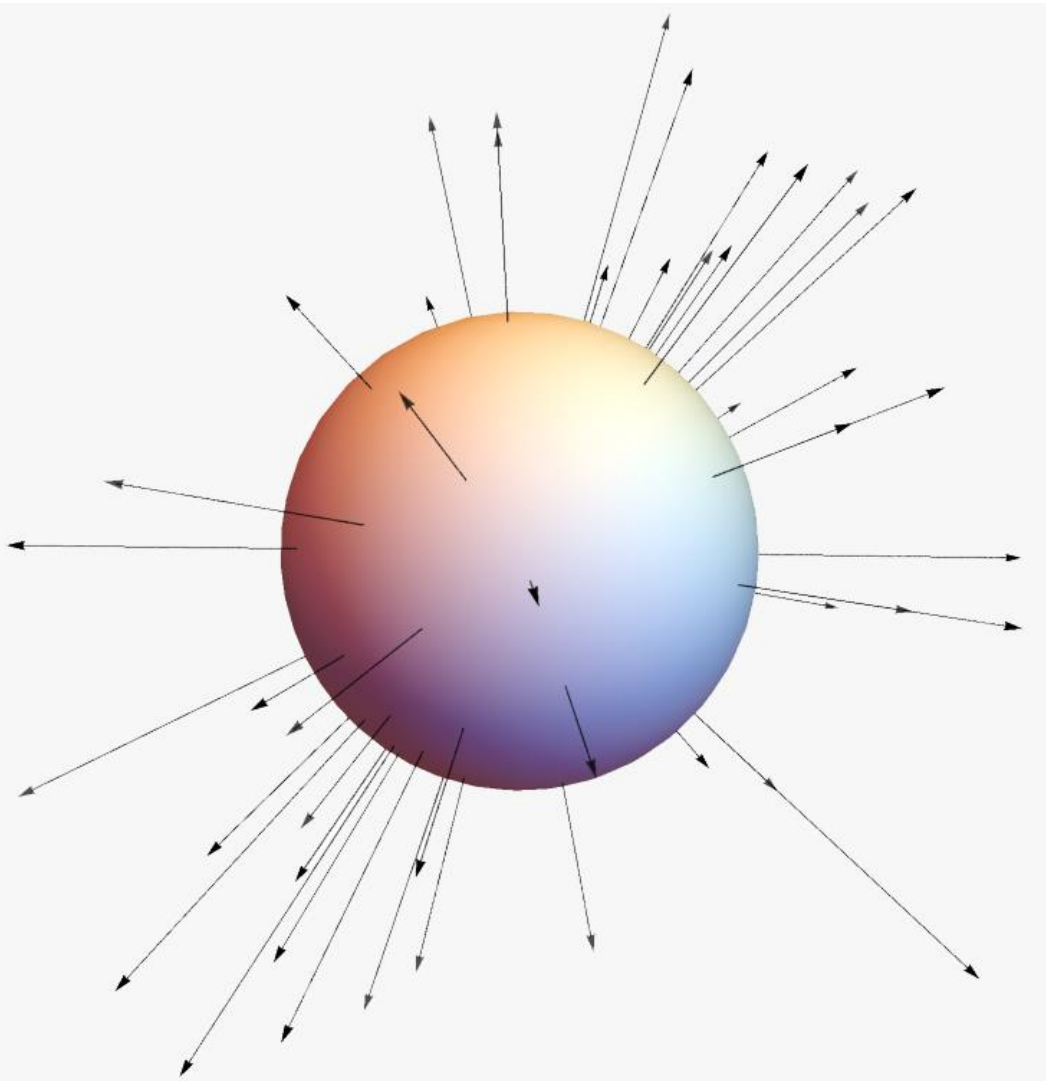
## ◆ LDA + Embedding?

- Gaussian LDA



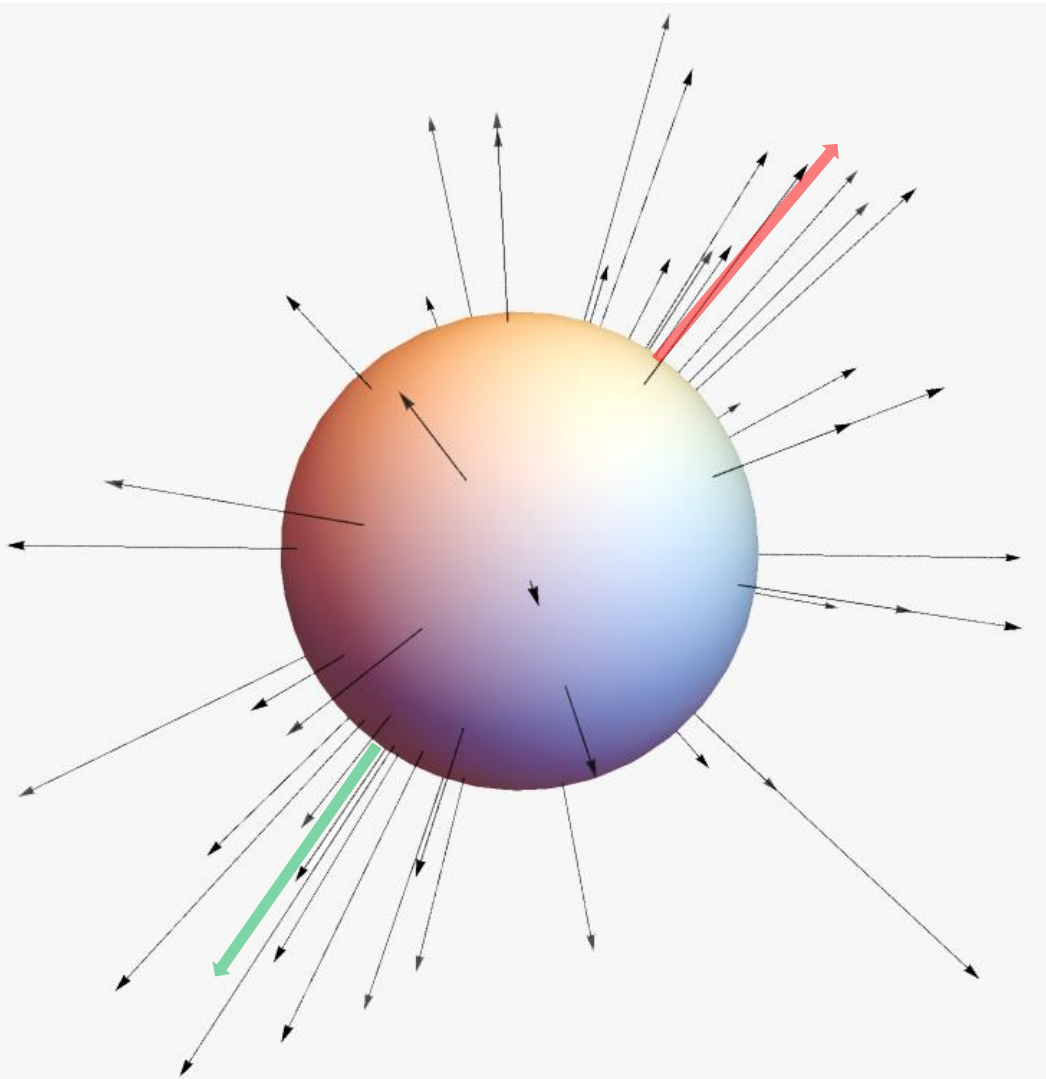
**Projection of the embedding vectors to 2-D**



# Bag-of-Vectors View of a Document



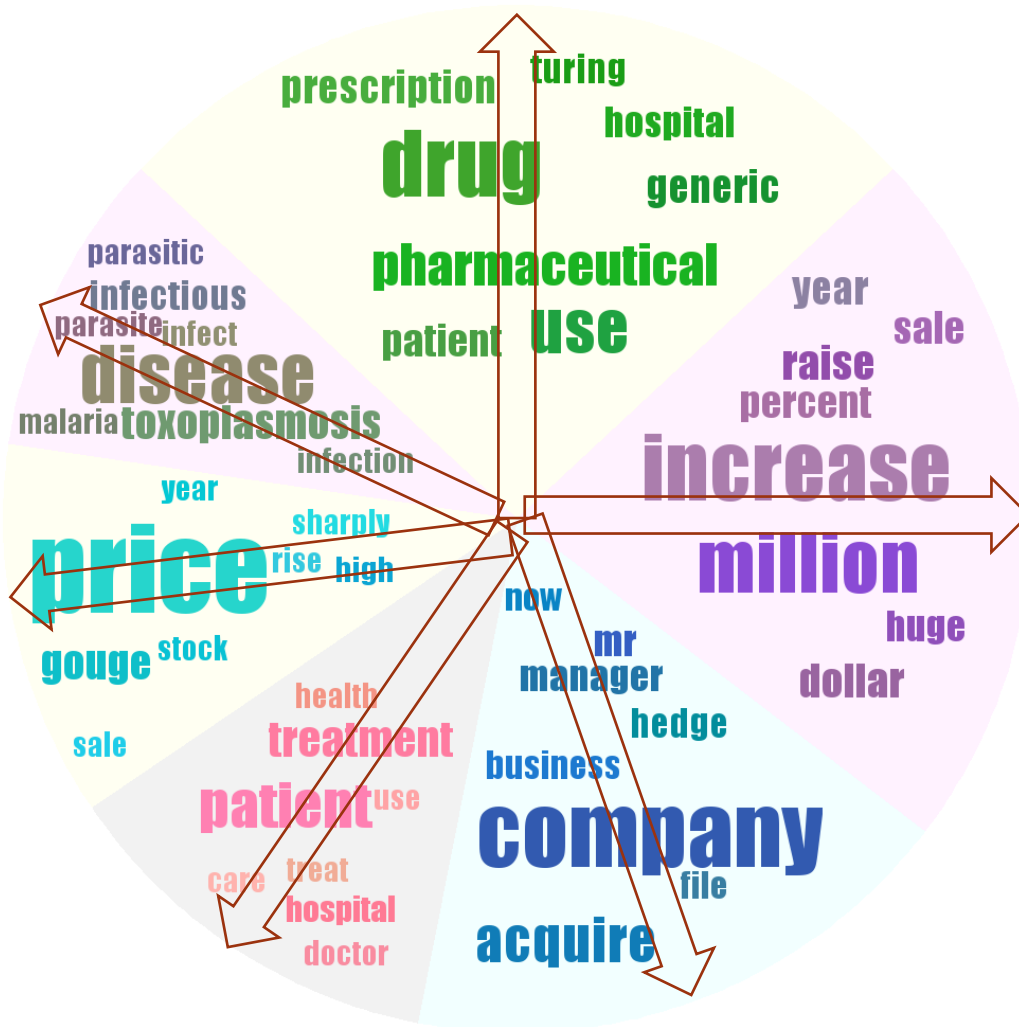
- ◆ Represent each word by its embedding vector
- ◆ A doc = a set of vectors shooting in different directions
- ◆ Vectors form clusters
- ◆ Directions containing more vectors  $\approx$  typical semantics

# Topic Embeddings in a Single Document



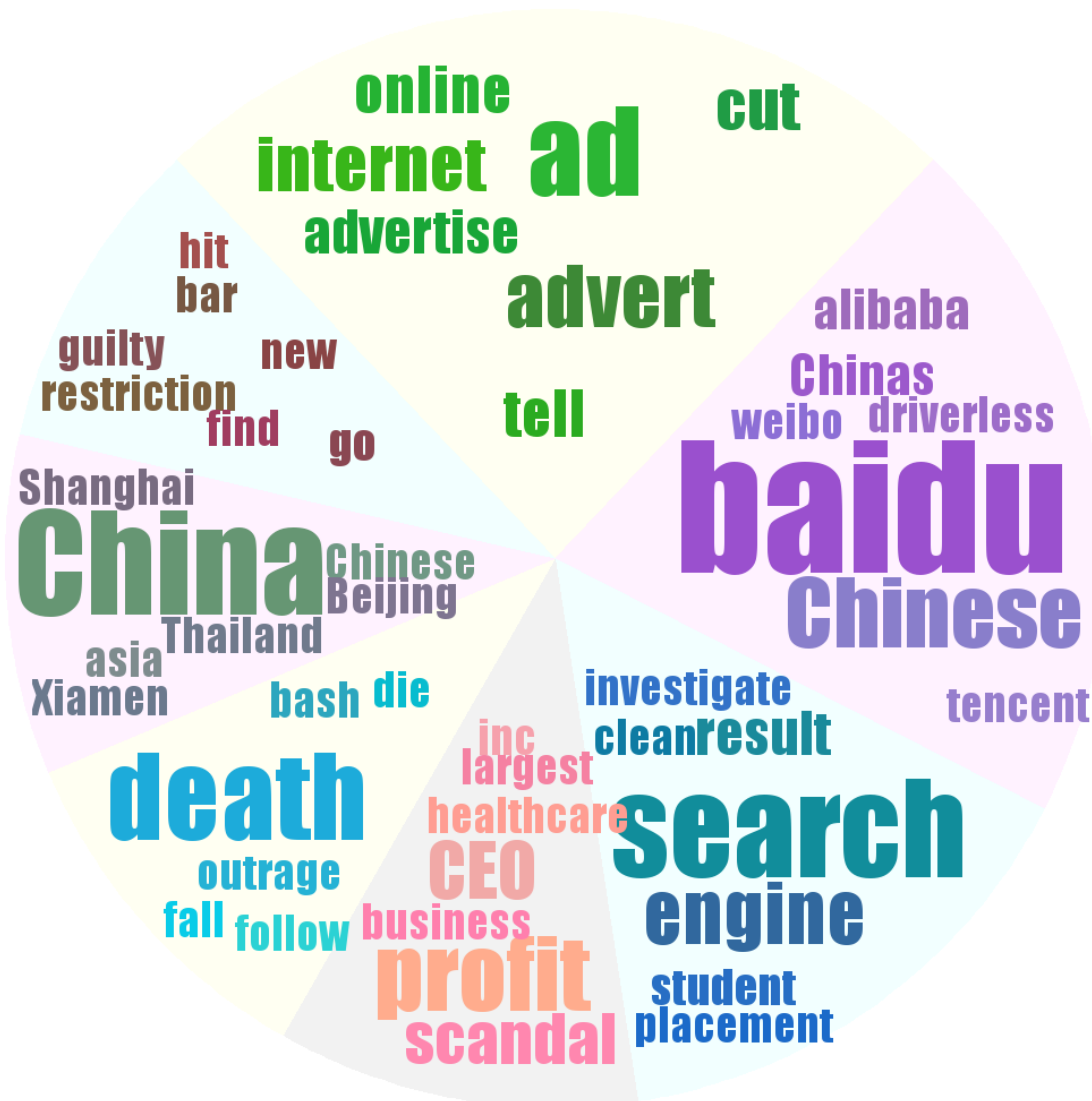
- ◆ Find a typical vector for a cluster of vectors
- ◆ Left graph: two typical vectors  , 
- ◆ **Typical vectors**  $\approx$  major semantics = **Topics**
- ◆ **Topic embeddings** represent the document concisely
- ◆ Minor semantics are ignored

# Topic Embeddings Illustrated



◆ On a news article about a pharmaceutical company acquisition

# Topic Embeddings Illustrated (2)



◆ 16944 tweets after  
Baidu Ads scandal  
(Wei Zexi incident)

# The Topic in a Topic Model

## ◆ Topic (a.k.a. **Topic-word distribution**)

- Each word in a document has a latent topic
- Each topic is a categorical distribution of words

$$p(w_1|\mathbf{t}_k) = p_{k1}, \dots, p(w_W|\mathbf{t}_k) = p_{kW}$$

- Most topics may have some vague semantics
  - Sports, electronics, computer science...
- Semantically more related words have higher prob

## ◆ Disadvantage

- Each word needs a probability as a parameter
- $K*(W-1)$  parameters — too many!

# Continuous Representation of Topics

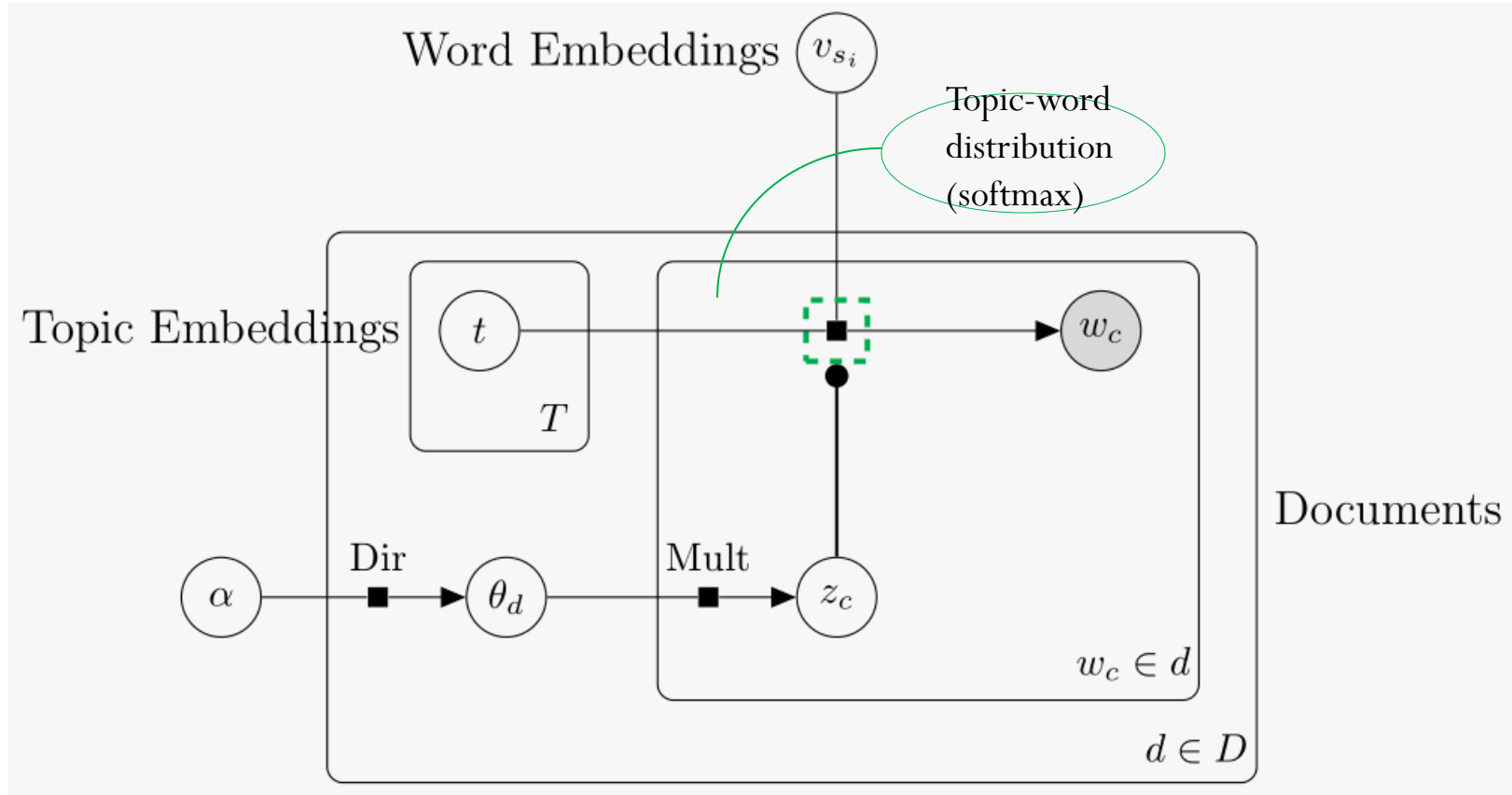
- ◆ Probability of each word  $w_i$  under topic  $k$ :

$$p(w_i|\mathbf{t}_k) = \frac{P(w_i) \exp\{\mathbf{v}_{w_i}^T \mathbf{t}_k\}}{\sum_j P(w_j) \exp\{\mathbf{v}_{w_j}^T \mathbf{t}_k\}}$$

- $P(w_i)$ : prior prob of  $w_i$
- $\mathbf{v}_{w_i}$ : pretrained word embedding
- ◆ Predict a word probability by the dot-product of the topic embedding and the word embedding
  - Semantically more related words have higher prob
  - Ensures topic coherence
- ◆ Topic embedding:  $\mathbf{t}_k$  (100~500 dimensional)
  - Parameter number: 500\*K



# Graphical Model



◆ Major difference with LDA: formulation of topic-word distribution

# Evaluation on Document Classification

	20News			Reuters		
	Prec	Rec	F1	Prec	Rec	F1
BOW	69.1	68.5	68.6	92.5	90.3	91.1
LDA	61.9	61.4	60.3	76.1	74.3	74.8
sLDA	61.4	60.9	60.9	88.3	83.3	85.1
LFTM	63.5	64.8	63.7	84.6	86.3	84.9
MeanWV	70.4	70.3	70.1	92.0	89.6	90.5
Doc2Vec	56.3	56.6	55.4	84.4	50.0	58.5
TWE	69.5	69.3	68.8	91.0	89.1	89.9
GaussianLDA	30.9	26.5	22.7	46.2	31.5	35.3
TopicVec	71.3	71.3	71.2	<b>92.5</b>	<b>92.1</b>	<b>92.2</b>
TV+MeanWV	<b>71.8</b>	<b>71.5</b>	<b>71.6</b>	92.2	91.6	91.6

- ◆ It measures how much semantic info is preserved
- ◆ TopicVec: Topic proportions as features
- ◆ TopicVec & TV+WV are best
- ◆ MeanWV (mean word embedding) is not always helpful for TopicVec

Leverage deep neural networks ...

# Deep Generative Models

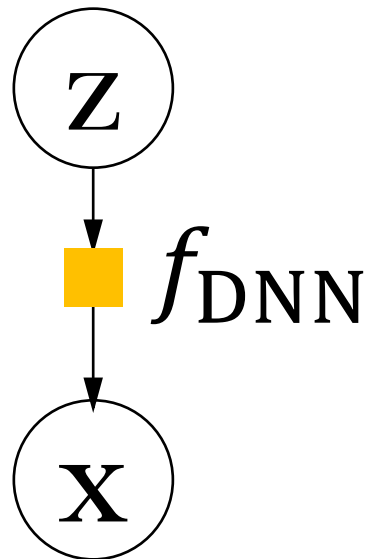
- ◆ If  $z$  is uniformly distributed over  $(0, 1)$ , then  $y = f(z)$  has the distribution

$$p(y) = p(z) \left| \frac{dz}{dy} \right|$$

- where  $p(z) = 1$
- ◆ This trick is widely used to draw samples from exponential family distributions (e.g., Gaussian, Exponential)
- ◆ *However, the function is typically simple. Can we learn it?*

# Deep Generative Models

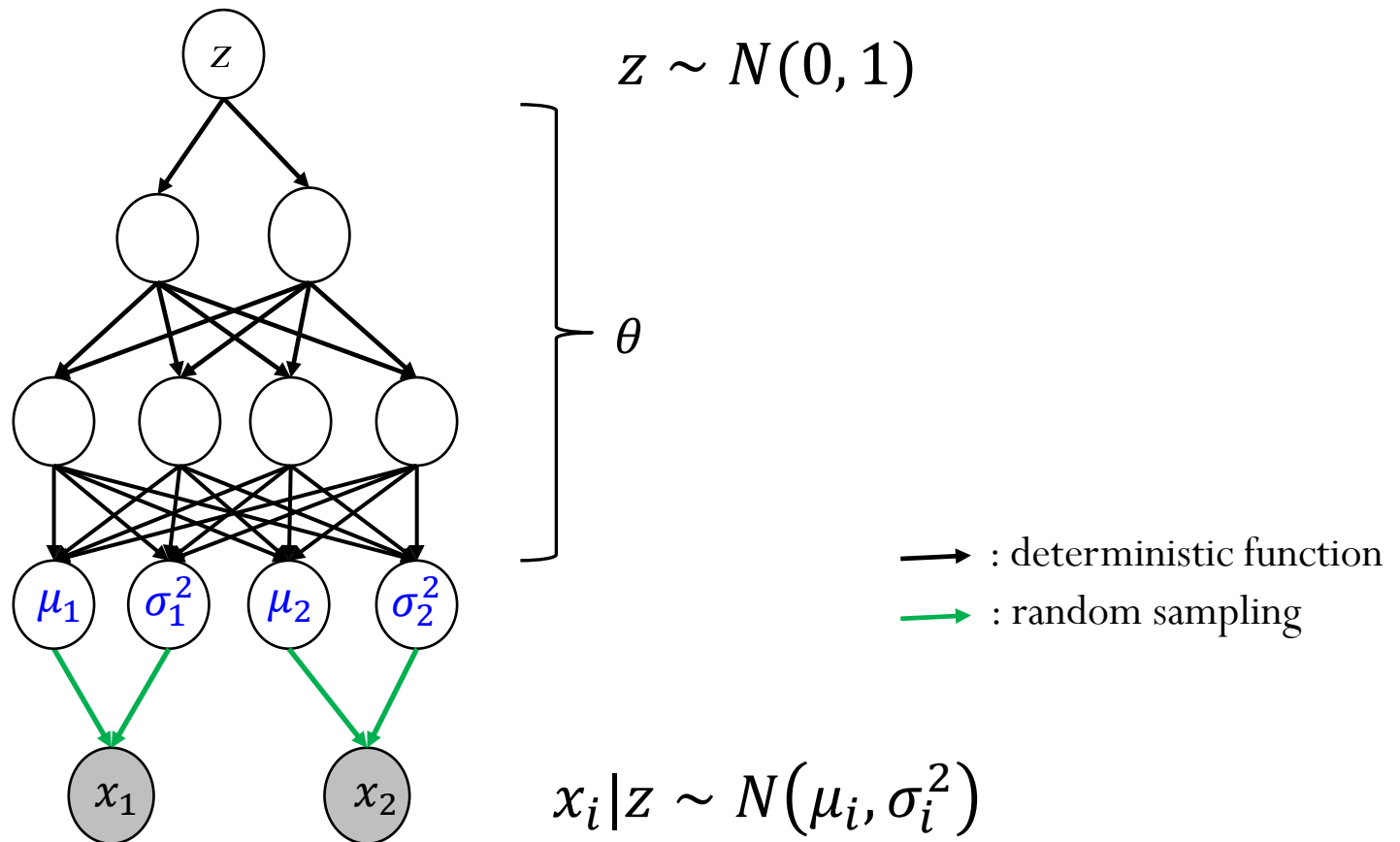
- ◆ More flexible by using differential function mapping between random variables
- ◆ DGMs learn a function transform with deep neural networks



Cause  $\Rightarrow$  Disease  
Topics  $\Rightarrow$  Docs  
Objects  $\Rightarrow$  Images  
Words  $\Rightarrow$  Phonemes

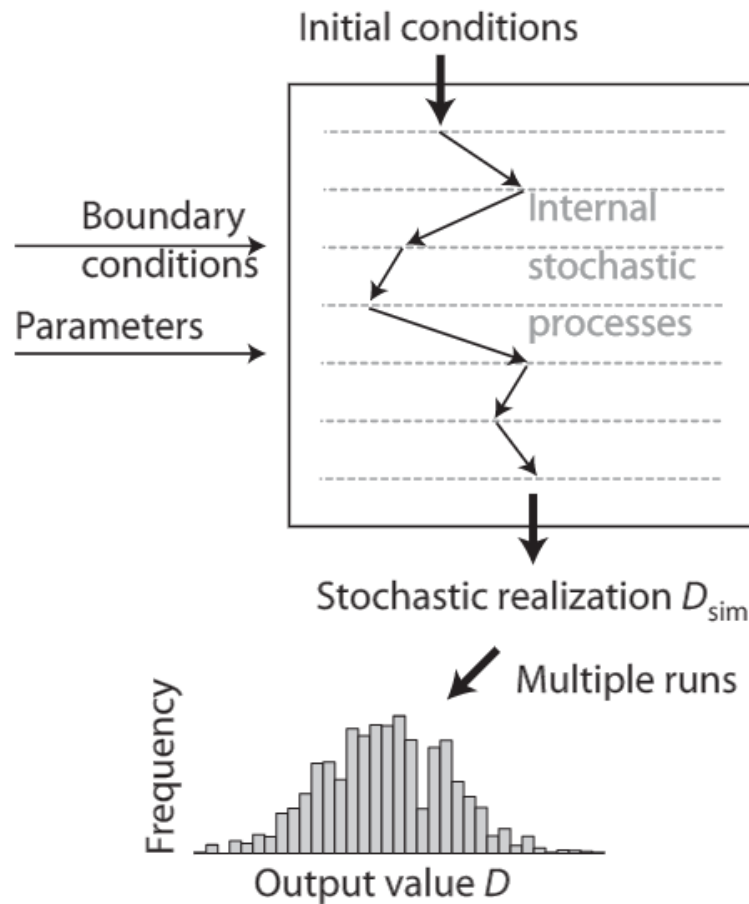
# An example with MLP

- ◆ 1D latent variable  $z$ ; 2D observation  $x$
- ◆ **Idea:** NN + Gaussian (or Bernoulli) with a diagonal covariance



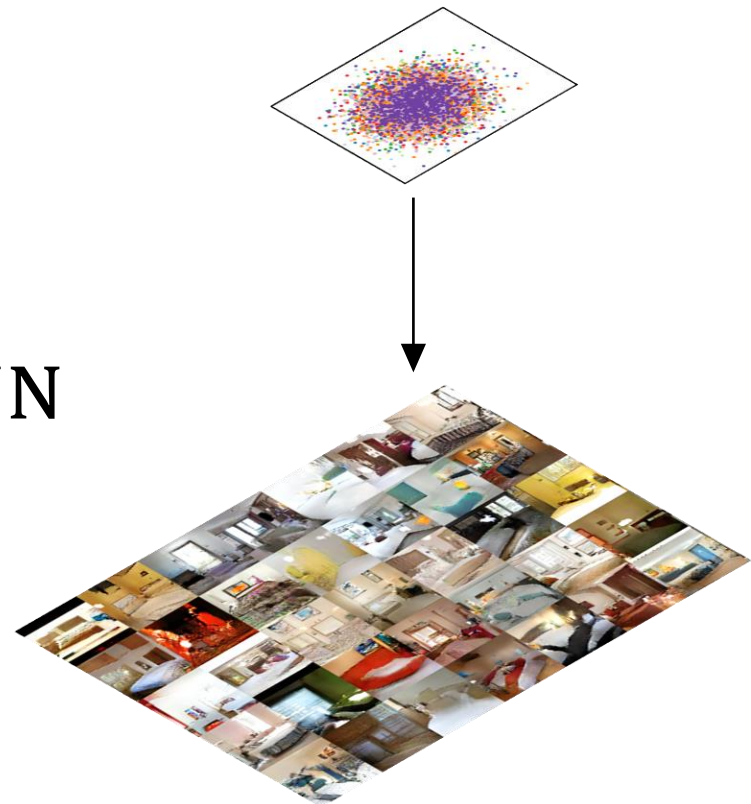
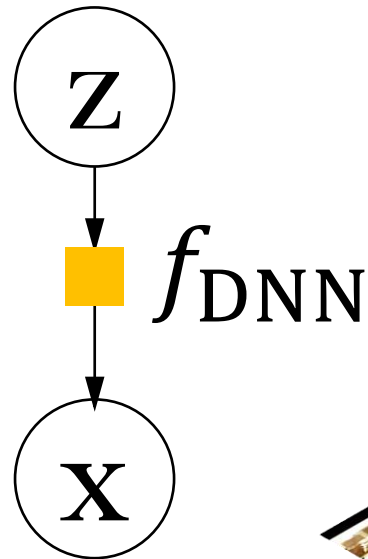
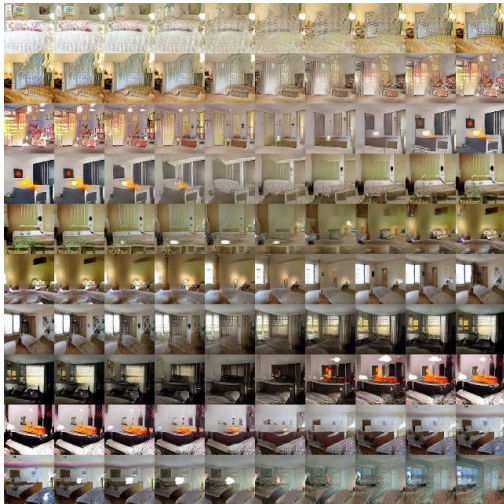
# Implicit Deep Generative Models

- ◆ Generate data with a stochastic process whose likelihood function is not explicitly specified (Hartig et al., 2011)



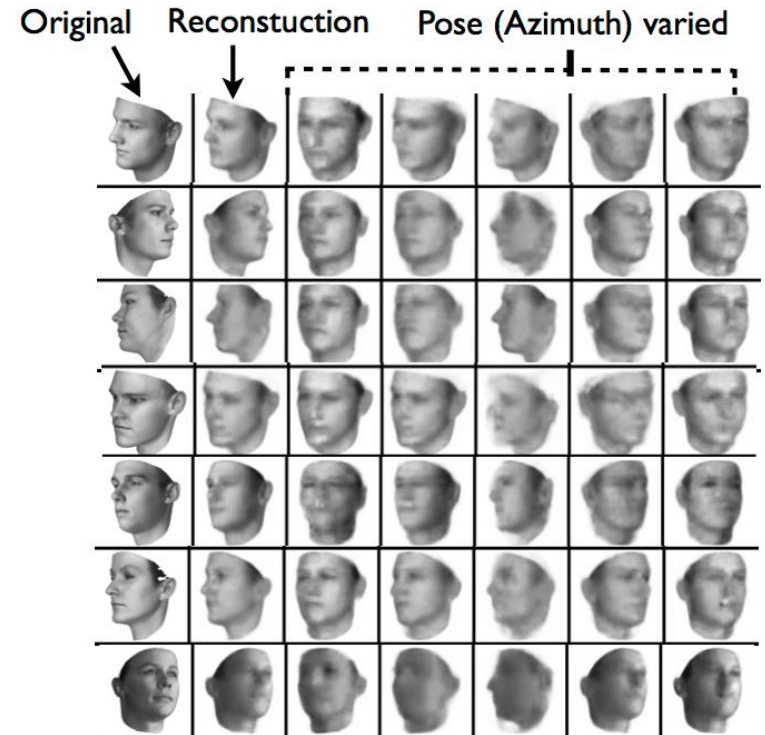
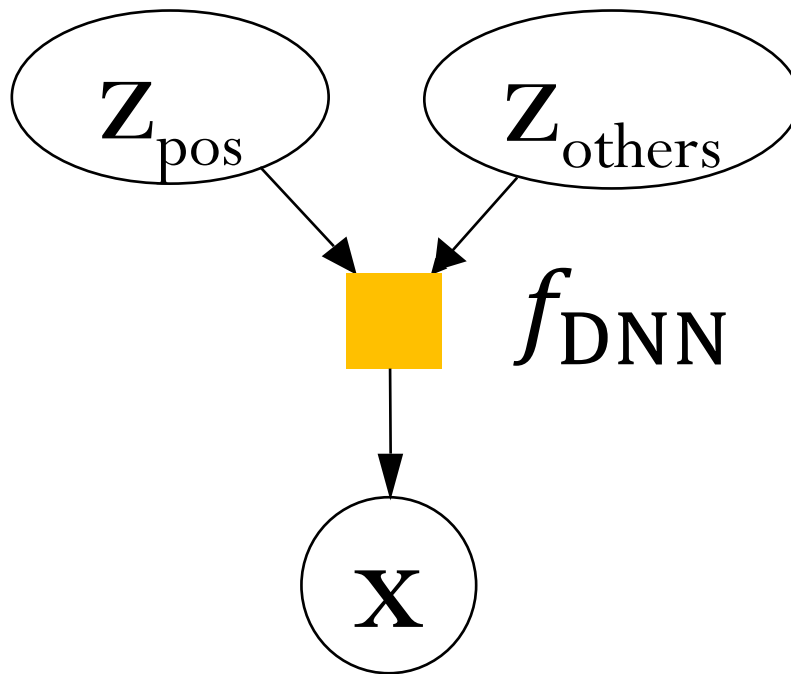
# Deep Generative Models

[Image Generation:  
Generative Adversarial Nets,  
Goodfellow13 & Radford15]





# Deep Generative Models

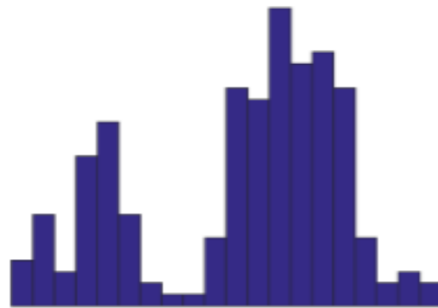


[Image Understanding: Variational Autoencoders,  
Kingma13 & Tejas15 & Eslami16]

# Learning Deep Generative Models

- ◆ **Given** a set  $D$  of unlabeled samples, learn the unknown parameters (or a distribution)

data  $D = \{x_i\}$



models  $p(x|\theta)$



- ◆ Find a model that minimizes

$$\mathbb{D}\left(\begin{array}{c} \text{data } \{x_i\}_{i=1}^n \\ \text{model } p \end{array}\right)$$

# Learning Deep Generative Models

- ◆ Maximum likelihood estimation (MLE):

$$\hat{\theta} = \operatorname{argmax} p(D|\theta)$$

- has an explicit likelihood model
- ◆ Minimax objective (e.g., GAN)
  - A two-player game to reach equilibrium
  - A three-player game for semi-supervised learning (NIPS'17)
- ◆ Moment-matching:
  - draw samples from  $p$ :  $\hat{D} = \{y_i\}_{i=1}^M$ , where  $y_i \sim p(x|\theta)$
  - Kernel MMD (NIPS'16):
    - rich enough to distinguish any two distributions in certain RKHS
  - PMD (NIPS'17)

# Variational Bayes

- ◆ Consider the log-likelihood of *a single example*

$$\log p(\mathbf{x}; \theta) = \log \int p(\mathbf{z}, \mathbf{x}; \theta) d\mathbf{z}$$

- ◆ Log-integral/sum is annoying to handle directly
- ◆ Derive a variational lower bound  $L(\theta, \phi, \mathbf{x})$

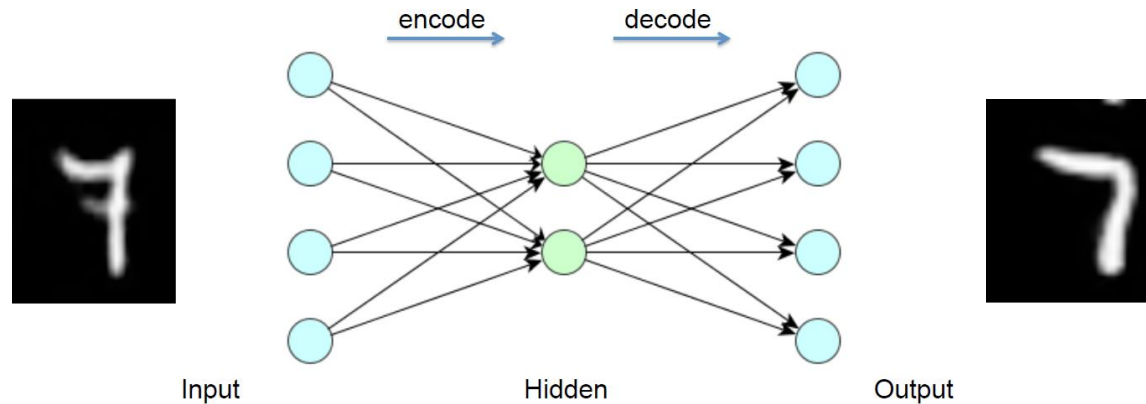
$$\log p(\mathbf{x}; \theta) = L(\theta, \phi, \mathbf{x}) + \text{KL}(q(\mathbf{z}|\mathbf{x}; \phi) \| p(\mathbf{z}|\mathbf{x}; \theta))$$

$$\begin{aligned} L(\theta, \phi, \mathbf{x}) &= \mathbf{E}_{q(\mathbf{z}|\mathbf{x}; \phi)} [\log p(\mathbf{z}, \mathbf{x}; \theta) - \log q(\mathbf{z}|\mathbf{x}; \phi)] \\ &= \mathbf{E}_{q(\mathbf{z}|\mathbf{x}; \phi)} [\log p(\mathbf{x}|\mathbf{z}; \theta) + \log p(\mathbf{z}; \theta) - \log q(\mathbf{z}|\mathbf{x}; \phi)] \\ &= \mathbf{E}_{q(\mathbf{z}|\mathbf{x}; \phi)} [\log p(\mathbf{x}|\mathbf{z}; \theta)] - \text{KL}(q(\mathbf{z}|\mathbf{x}; \phi) \| p(\mathbf{z}; \theta)) \end{aligned}$$

**reconstruction term**

**prior regularization**

# Recap: Auto-Encoder

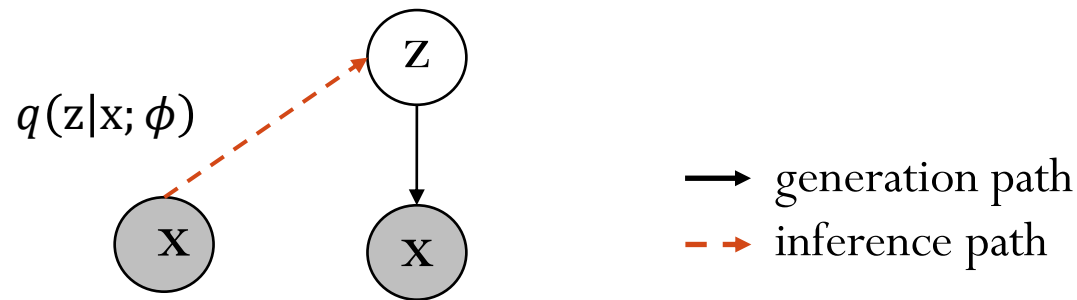


- ◆ Encoder:  $h = s(Wx + b)$
- ◆ Decoder:  $x' = s(W'h + b')$
- ◆ Training: minimize the reconstruction error (e.g., square loss, cross-entropy loss)
- ◆ Denoising AE: randomly corrupted inputs are restored to learn more robust features

# Auto-Encoding Variational Bayes (AEVB)

- ◆ What's unique in AEVB is that *the variational distribution is parameterized by a deep neural network*

$$q(z|x; \phi) \approx p(z|x; \theta)$$

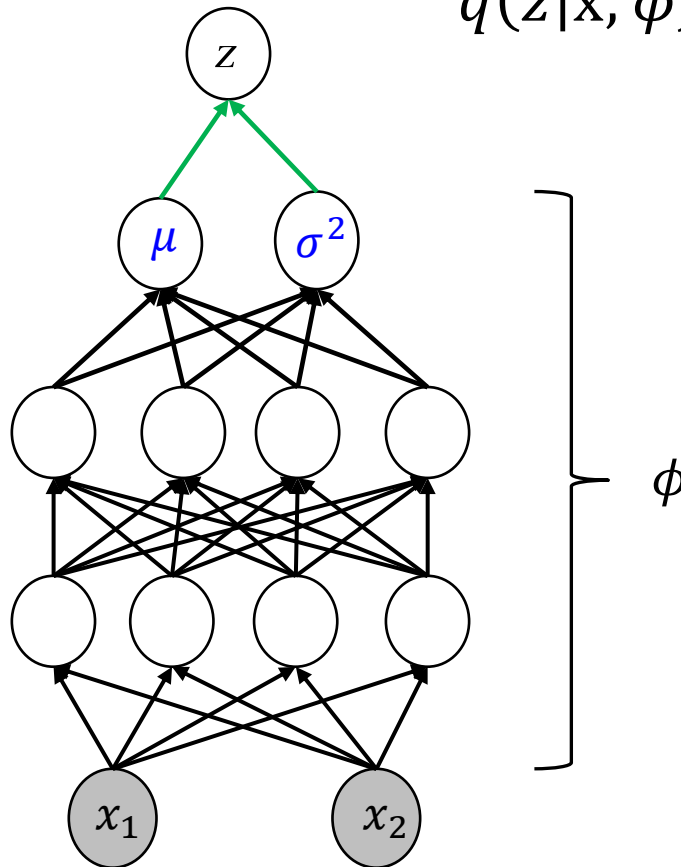


- We call it an **inference (recognition, encoder) network** or a **Q-network**
- All the parameters are learned jointly via SGD with variance reduction

# The Encoder Network

◆ A feedforward NN + Gaussian

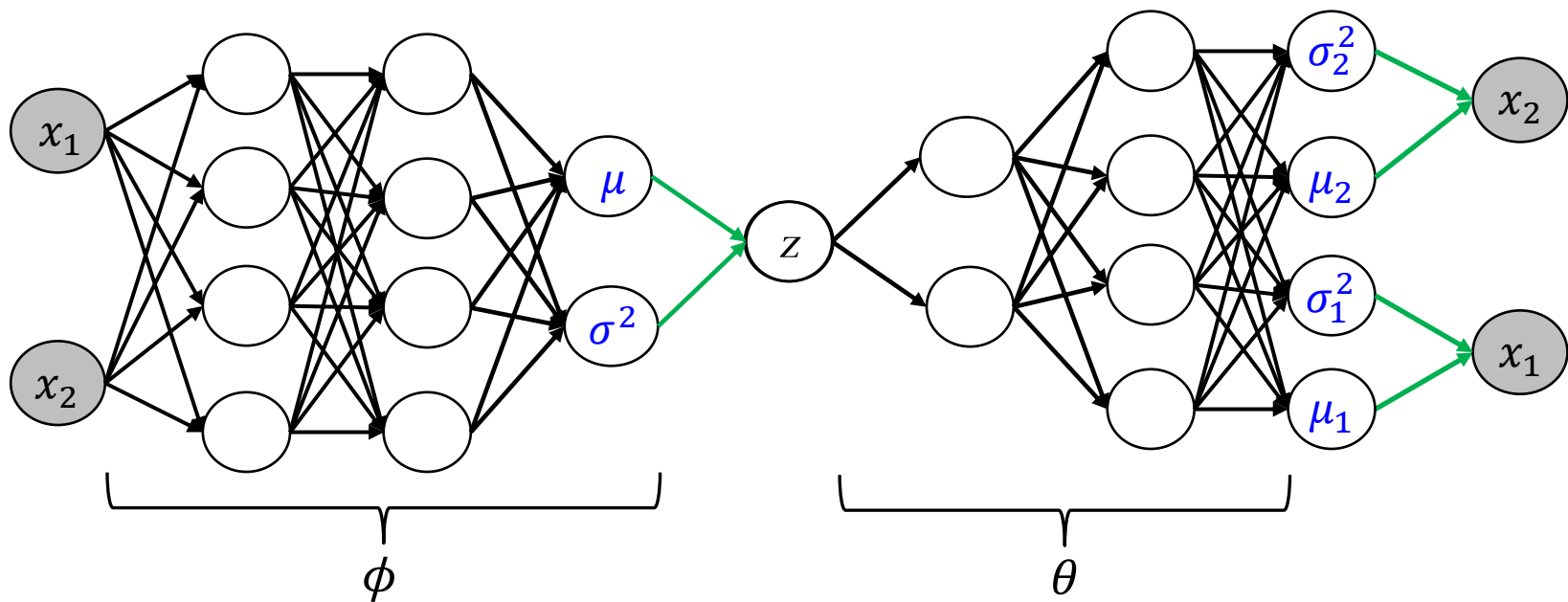
$$q(z|x; \phi) = N(\mu(x; \phi), \sigma^2(x; \phi))$$



→ : deterministic function  
→ : random sampling

# The Complete Auto-encoder

◆ The Q-P network architecture:



→ : deterministic function

→ : random sampling



# Stochastic Variational Inference

- ◆ Variational lower-bound for *a single example*

$$L(\theta, \phi, \mathbf{x}) = \mathbf{E}_{q(\mathbf{z}|\mathbf{x};\phi)}[\log p(\mathbf{x}|\mathbf{z}; \theta)] - \text{KL}(q(\mathbf{z}|\mathbf{x}; \phi) \| p(\mathbf{z}; \theta))$$

- ◆ Variational lower-bound for *a set of examples*

$$L(\theta, \phi, D) = \sum_i \mathbf{E}_{q(\mathbf{z}_i|\mathbf{x}_i;\phi)}[\log p(\mathbf{x}_i|\mathbf{z}_i; \theta)] - \text{KL}(q(\mathbf{z}_i|\mathbf{x}_i; \phi) \| p(\mathbf{z}_i; \theta))$$

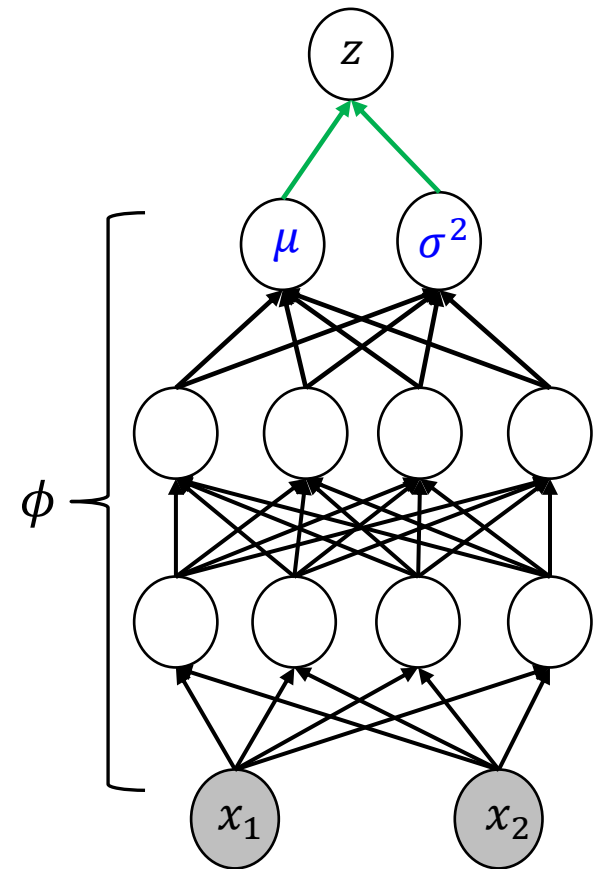
- Use stochastic gradient methods to handle large datasets
- Random mini-batch
  - for each  $i$ , infer the posterior  $q(\mathbf{z}_i|\mathbf{x}_i; \phi)$  ; As we parameterize as a neural network, this in fact optimizes  $\phi$
- ◆ *However, calculating the expectation and its gradients is non-trivial, often intractable*

# Example with Gaussian Distributions

- ◆ Use  $N(0, 1)$  as prior for  $z$ ;  $q(z|x; \phi)$  is Gaussian with parameters  $(\mu(x; \phi), \sigma^2(x; \phi))$  determined by NN
  - The KL-divergence

$$-\text{KL}(q(z|x; \phi) \| p(z; \theta)) = \frac{1}{2} (1 + \log \sigma^2 - \mu^2 - \sigma^2)$$

- **Homework:** finish the derivation



# Example with Gaussian Distributions

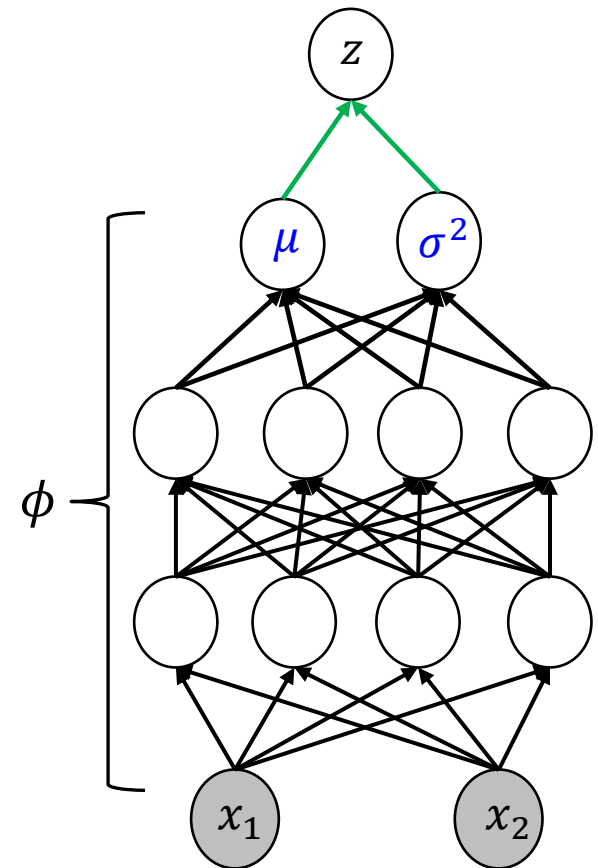
- ◆ Use  $N(0, 1)$  as prior for  $z$ ;  $q(z|x; \phi)$  is Gaussian with parameters  $(\mu(x; \phi), \sigma^2(x; \phi))$  determined by NN
  - The expected log-likelihood

$$\mathbf{E}_{q(z|x; \phi)} [\log p(x|z; \theta)]$$

- If the likelihood is Gaussian

$$-\log p(x_i|z_i) = \sum_j \frac{1}{2} \log \sigma_j^2 + \frac{(x_{ij} - \mu_{xi})^2}{2\sigma_j^2}$$

- *The expectation is still hard to compute because of nonlinearity functions*



# Example with Gaussian Distributions

- ◆ Use  $N(0, 1)$  as prior for  $z$ ;  $q(z|x; \phi)$  is Gaussian with parameters  $(\mu(x; \phi), \sigma^2(x; \phi))$  determined by NN
  - The expected log-likelihood

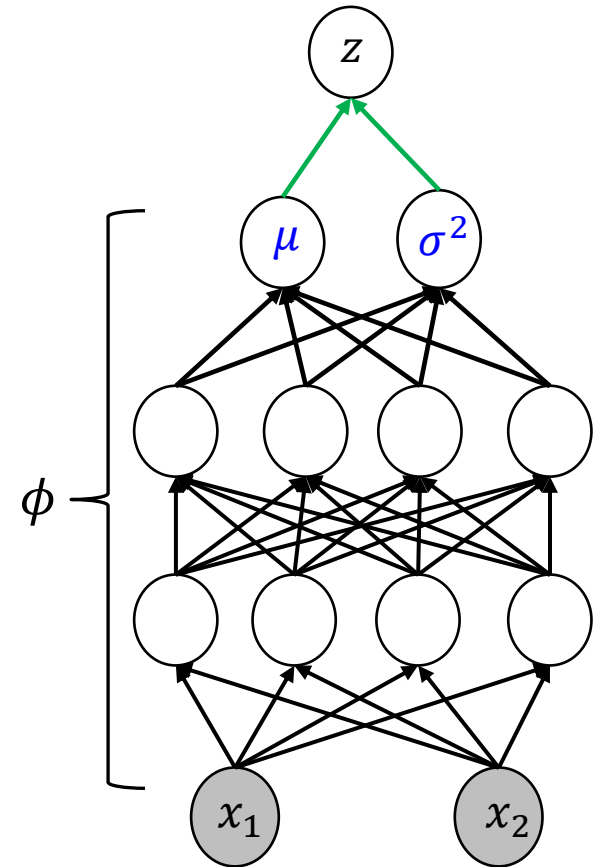
$$\mathbf{E}_{q(z|x; \phi)} [\log p(x|z; \theta)]$$

- Approximate via Monte Carlo methods

$$\mathbf{E}_{q(z|x; \phi)} [\log p(x|z; \theta)] \approx \frac{1}{L} \sum_k \log p(x|z^{(k)})$$

$$z^{(k)} \sim q(z|x; \phi)$$

- An unbiased estimator



# Example with Gaussian Distributions

- ◆ The KL-regularization term (closed-form):

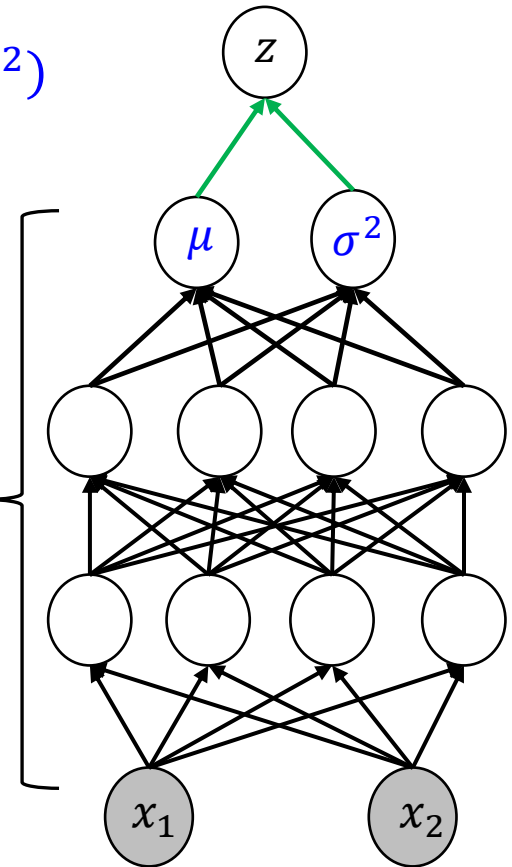
$$-\text{KL}(q(z|x; \phi) \| p(z; \theta)) = \frac{1}{2} (1 + \log \sigma^2 - \mu^2 - \sigma^2)$$

- Easy to calculate gradient
- ◆ The expected log-likelihood term (MC estimate)

$$\mathbf{E}_{q(z|x; \phi)} [\log p(x|z; \theta)] \approx \frac{1}{L} \sum_k \log p(x|z^{(k)})$$

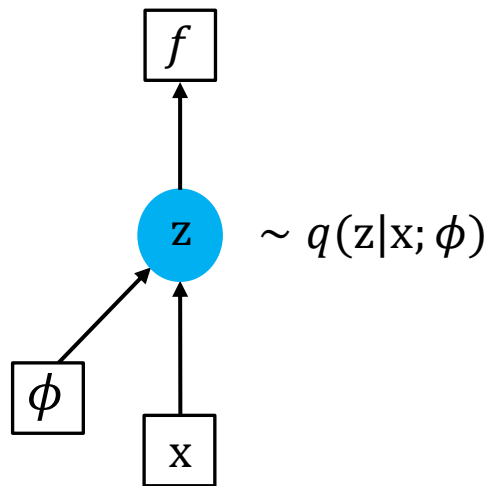
$$z^{(k)} \sim q(z|x; \phi)$$

- Gradient needs back-propagation!
- *However,  $z^{(k)}$  is a random variable, we can't take gradient over a randomly drawn number*



# Reparameterization Trick

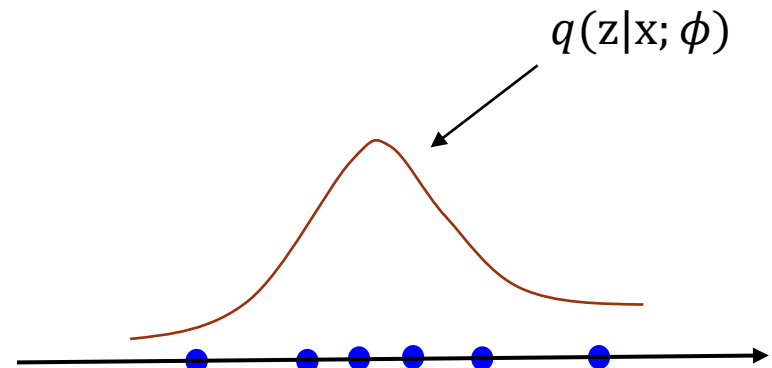
◆ Backpropagation not possible through random sampling



□ : deterministic node    ● : random node

$$z^{(k)} \sim N(\mu(x, \phi), \sigma^2(x, \phi))$$

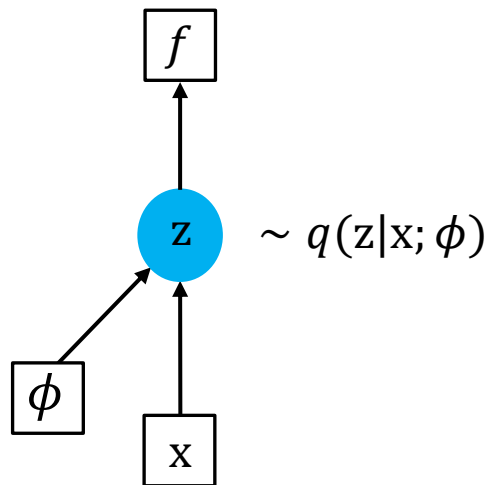
Cannot back-propagate through a  
randomly drawn number



$\{-1.5, -0.5, 0.3, 0.6, 1.5, \dots\}$

# Reparameterization Trick

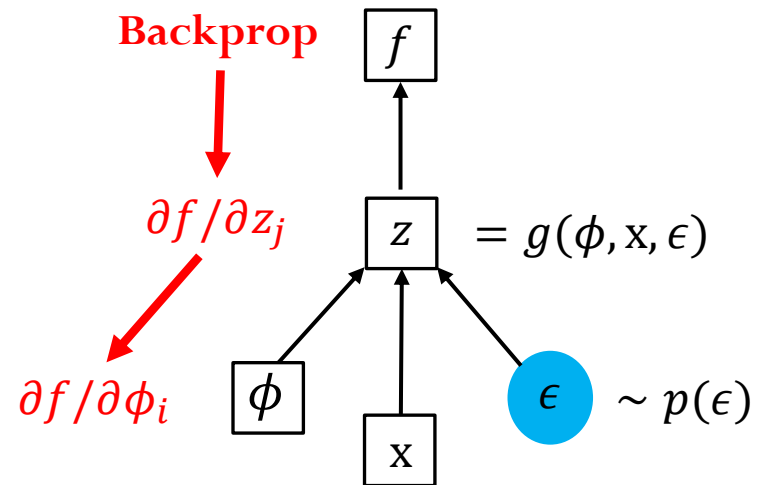
◆ Backpropagation not possible through random sampling



□: deterministic node    ●: random node

$$z^{(k)} \sim N(\mu(x, \phi), \sigma^2(x, \phi))$$

Cannot back-propagate through a randomly drawn number



$$\epsilon^{(k)} \sim N(0,1)$$

$$z^{(k)} = \mu(x, \phi) + \sigma(x, \phi) \cdot \epsilon^{(k)}$$

$z$  has the same distribution, but now can back-prop  
Separate into a deterministic part and noise

# The General Form

◆ The VAE bound

$$\begin{aligned} L(\theta, \phi, \mathbf{x}) &= \mathbf{E}_{q(\mathbf{z}|\mathbf{x};\phi)} [\log p(\mathbf{z}, \mathbf{x}; \theta) - \log q(\mathbf{z}|\mathbf{x}; \phi)] \\ &= \mathbf{E}_{q(\mathbf{z}|\mathbf{x};\phi)} \left[ \log \frac{p(\mathbf{z}, \mathbf{x}; \theta)}{q(\mathbf{z}|\mathbf{x}; \phi)} \right] \end{aligned}$$

◆ Monte Carlo estimate:

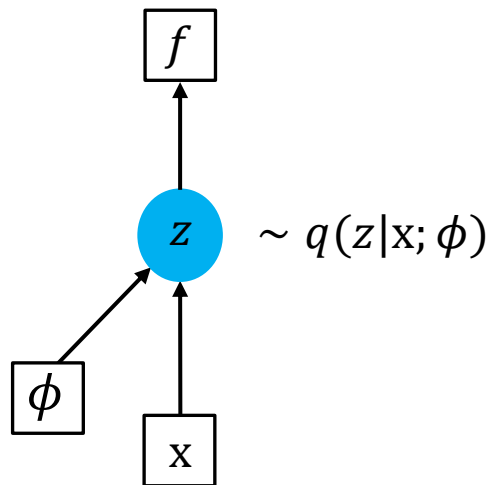
$$\begin{aligned} L(\theta, \phi, \mathbf{x}) &\approx \frac{1}{L} \sum_k \log \frac{p(\mathbf{z}^{(k)}, \mathbf{x}; \theta)}{q(\mathbf{z}^{(k)}|\mathbf{x}; \phi)} \\ \mathbf{z}^{(k)} &\sim q(\mathbf{z}|\mathbf{x}; \phi) \end{aligned}$$

- Again, we cannot back-prop through the randomly drawn numbers



# Reparameterization Trick

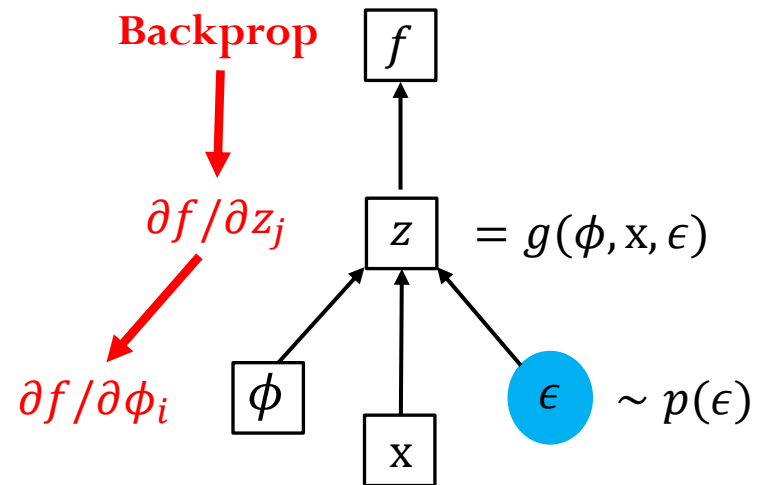
❖ Backpropagation not possible through random sampling



□: deterministic node    ●: random node

$$z^{(k)} \sim q(z|x; \phi)$$

Cannot back-propagate through a randomly drawn number



$$\epsilon^{(k)} \sim p(\epsilon)$$

$$z^{(k)} = g(\phi, x, \epsilon^{(k)})$$

$z$  has the same distribution, but now can back-prop  
Separate into a deterministic part and noise

# Reparam-Trick Summary

◆ The VAE bound

$$L(\theta, \phi, \mathbf{x}) = \mathbf{E}_{q(\mathbf{z}|\mathbf{x};\phi)} \left[ \log \frac{p(\mathbf{z}, \mathbf{x}; \theta)}{q(\mathbf{z}|\mathbf{x}; \phi)} \right]$$

□ Reparameterized as

$$L(\theta, \phi, \mathbf{x}) = \mathbf{E}_{p(\epsilon)} \left[ \log \frac{p(g(\mathbf{x}, \epsilon, \phi), \mathbf{x}; \theta)}{q(g(\mathbf{x}, \epsilon, \phi)|\mathbf{x}; \phi)} \right]$$

- where  $\epsilon$  is a simple distribution (e.g., standard normal) and  $g$  is a deep NN

◆ The gradients are

$$\nabla_{\theta} L(\theta, \phi, \mathbf{x}) = \mathbf{E}_{p(\epsilon)} \left[ \nabla_{\theta} \log \frac{p(g(\mathbf{x}, \epsilon, \phi), \mathbf{x}; \theta)}{q(g(\mathbf{x}, \epsilon, \phi)|\mathbf{x}; \phi)} \right]$$

- Back-prop is applied over the deep NN
- Similar for  $\phi$

# Importance Weighted Auto-Encoder (IWAE)

- ◆ The VAE lower bound of log-likelihood

$$L(\theta, \phi, \mathbf{x}) = \mathbf{E}_{q(\mathbf{z}|\mathbf{x};\phi)} \left[ \log \frac{p(\mathbf{z}, \mathbf{x}; \theta)}{q(\mathbf{z}|\mathbf{x}; \phi)} \right]$$

- ◆ A better variational lower bound (IWAE)

$$L_K(\theta, \phi, \mathbf{x}) = \mathbf{E}_{q(\mathbf{z}|\mathbf{x};\phi)} \left[ \log \left( \frac{1}{K} \sum_{k=1:K} \frac{p(\mathbf{z}^{(k)}, \mathbf{x}; \theta)}{q(\mathbf{z}^{(k)}|\mathbf{x}; \phi)} \right) \right]$$

where  $\mathbf{z}^{(k)} \sim q(\mathbf{z}|\mathbf{x}; \phi)$

- This is a lower-bound of the log-likelihood
- When  $K=1$ , recovers the VAE bound
- When  $K = \infty$ , recovers the log-likelihood
- A monotonic sequence:

$$L_K(\theta, \phi, \mathbf{x}) \leq L_{K+1}(\theta, \phi, \mathbf{x}), \quad \forall \theta, \phi, \mathbf{x}$$

# Reparametrization Trick

◆ The IWAE bound:

$$L_K(\theta, \phi, \mathbf{x}) = \mathbf{E}_{q(\mathbf{z}|\mathbf{x};\phi)} \left[ \log \left( \frac{1}{K} \sum_{k=1:K} w(\mathbf{z}^{(k)}, \mathbf{x}; \theta) \right) \right]$$

$$\text{where } \mathbf{z}^{(k)} \sim q(\mathbf{z}|\mathbf{x}; \phi) \quad w(\mathbf{z}^{(k)}, \mathbf{x}; \theta, \phi) = \frac{p(\mathbf{z}^{(k)}, \mathbf{x}; \theta)}{q(\mathbf{z}^{(k)}|\mathbf{x}; \phi)}$$

◆ Reparameterization form:

$$L_K(\theta, \phi, \mathbf{x}) = \mathbf{E}_{p(\epsilon)} \left[ \log \left( \frac{1}{K} \sum_{k=1:K} w(g(\epsilon^{(k)}, \mathbf{x}, \phi), \mathbf{x}; \theta) \right) \right]$$

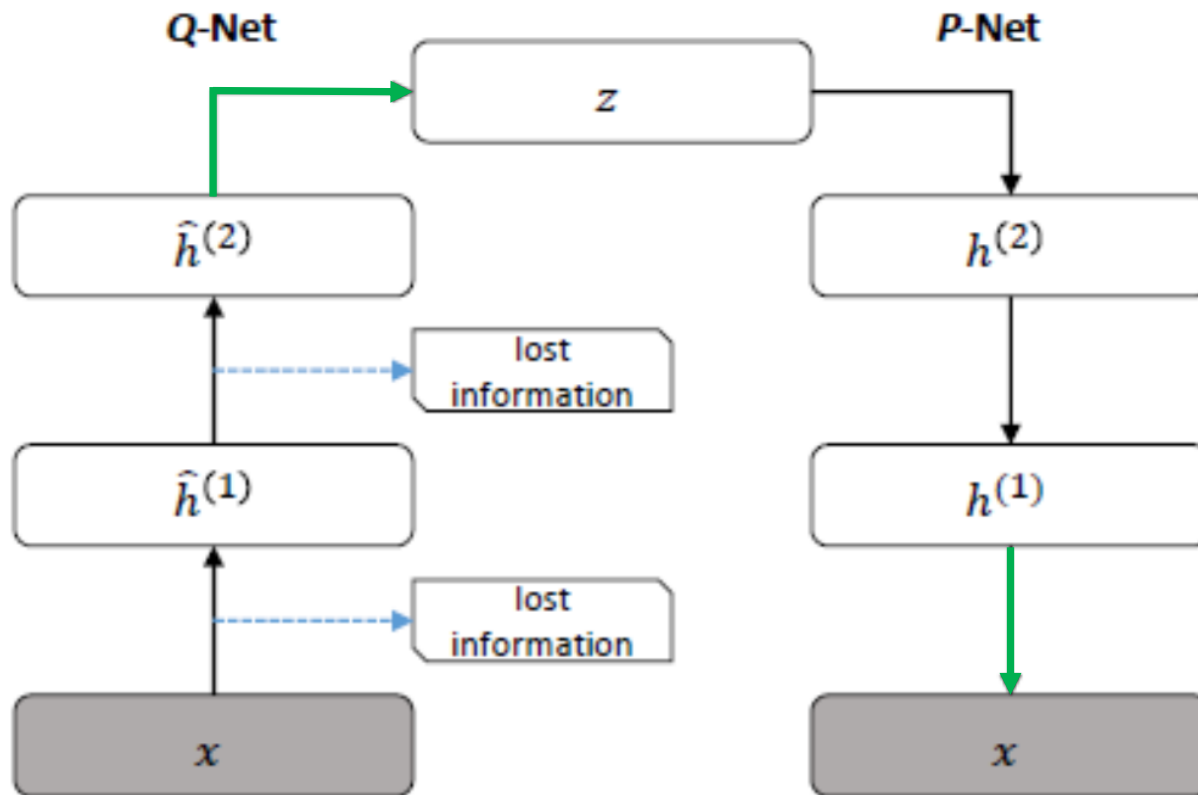
$$\text{where } \epsilon^{(k)} \sim p(\epsilon)$$

- The gradient can be calculated as in VAE

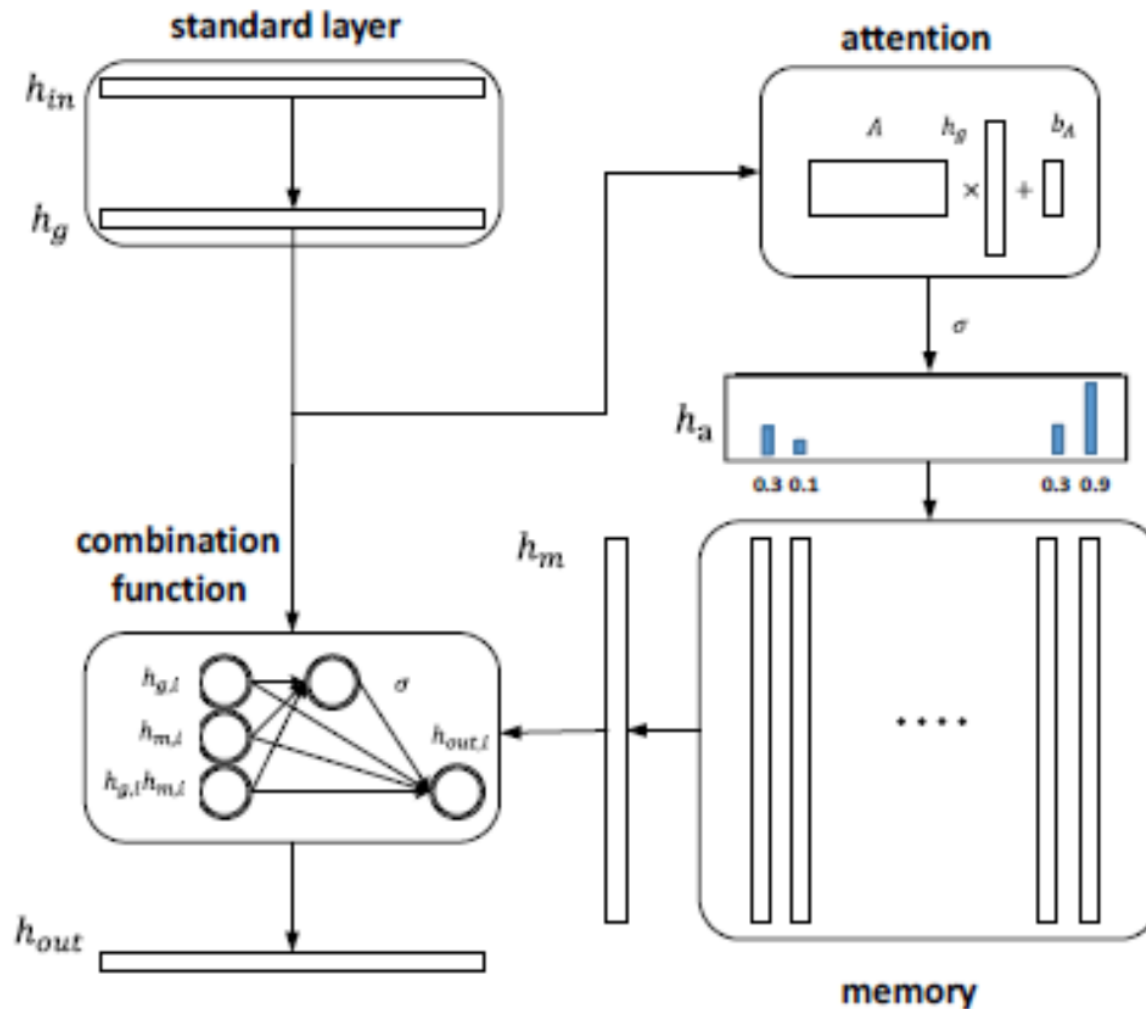
More Applications

# Symmetric Q-P Network

◆ **Problem:** detail information is lost during abstraction

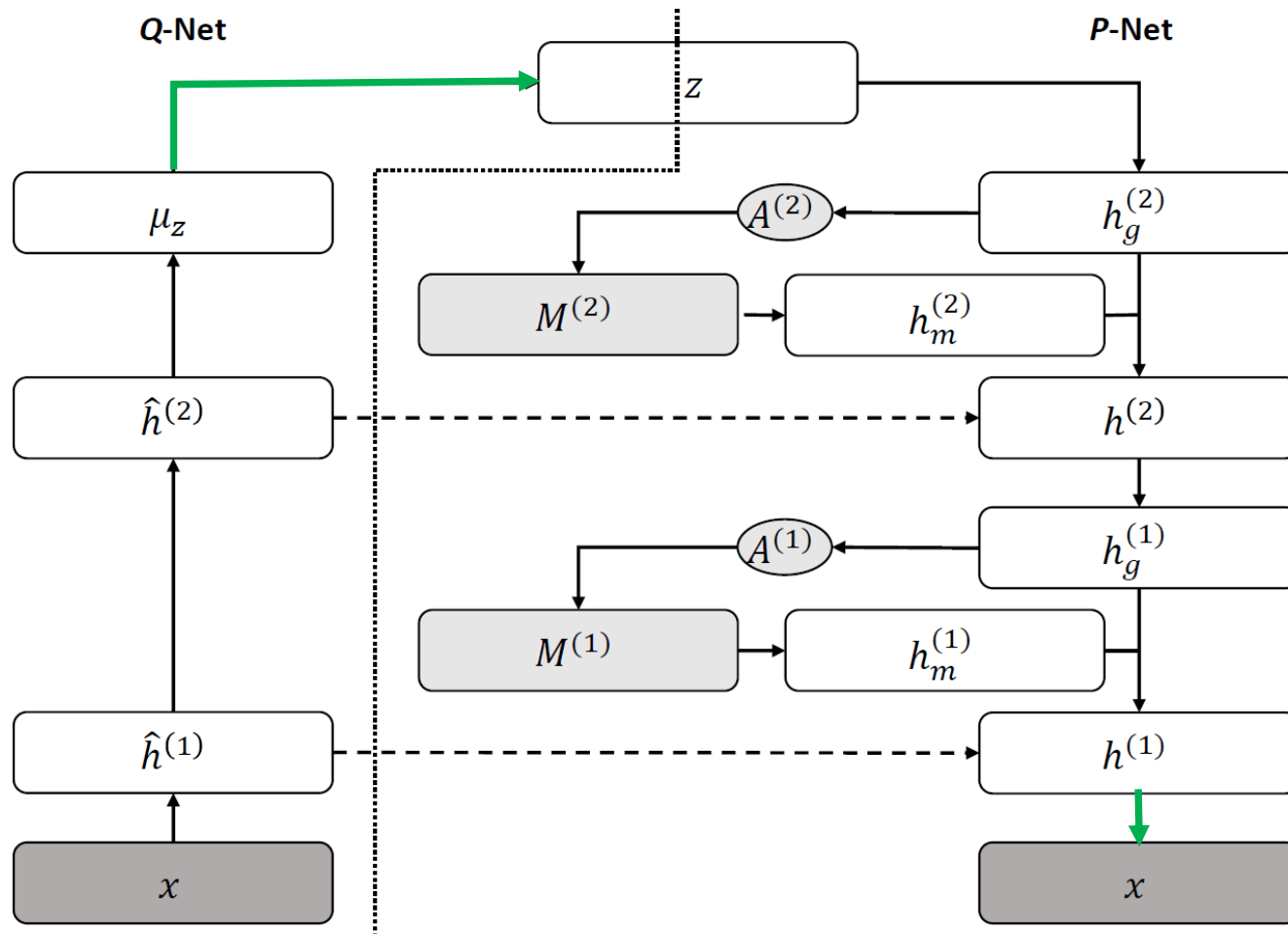


# A Layer with Memory and Attention



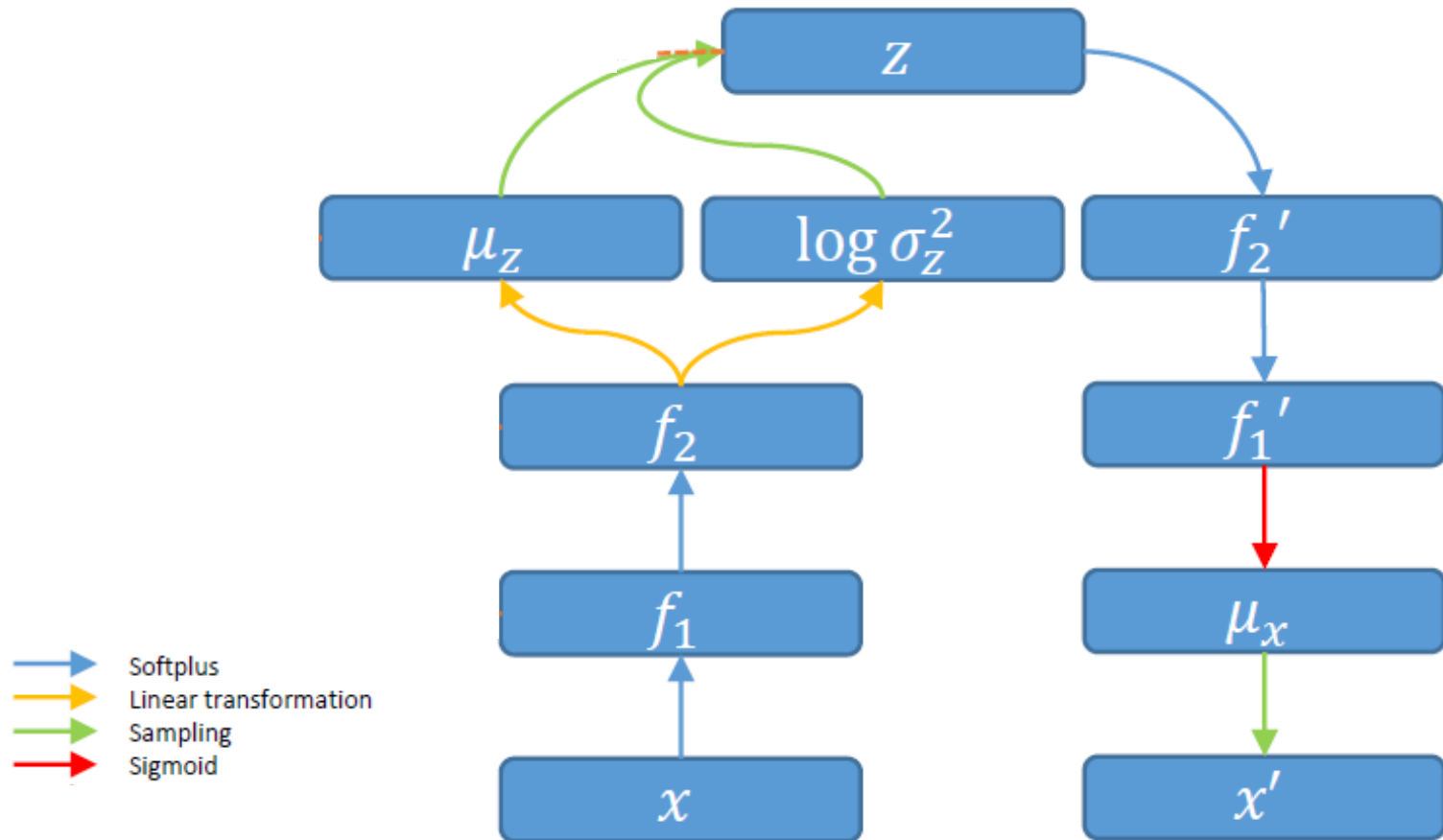
# A Stacked Deep Model with Memory

◆ Asymmetric architecture



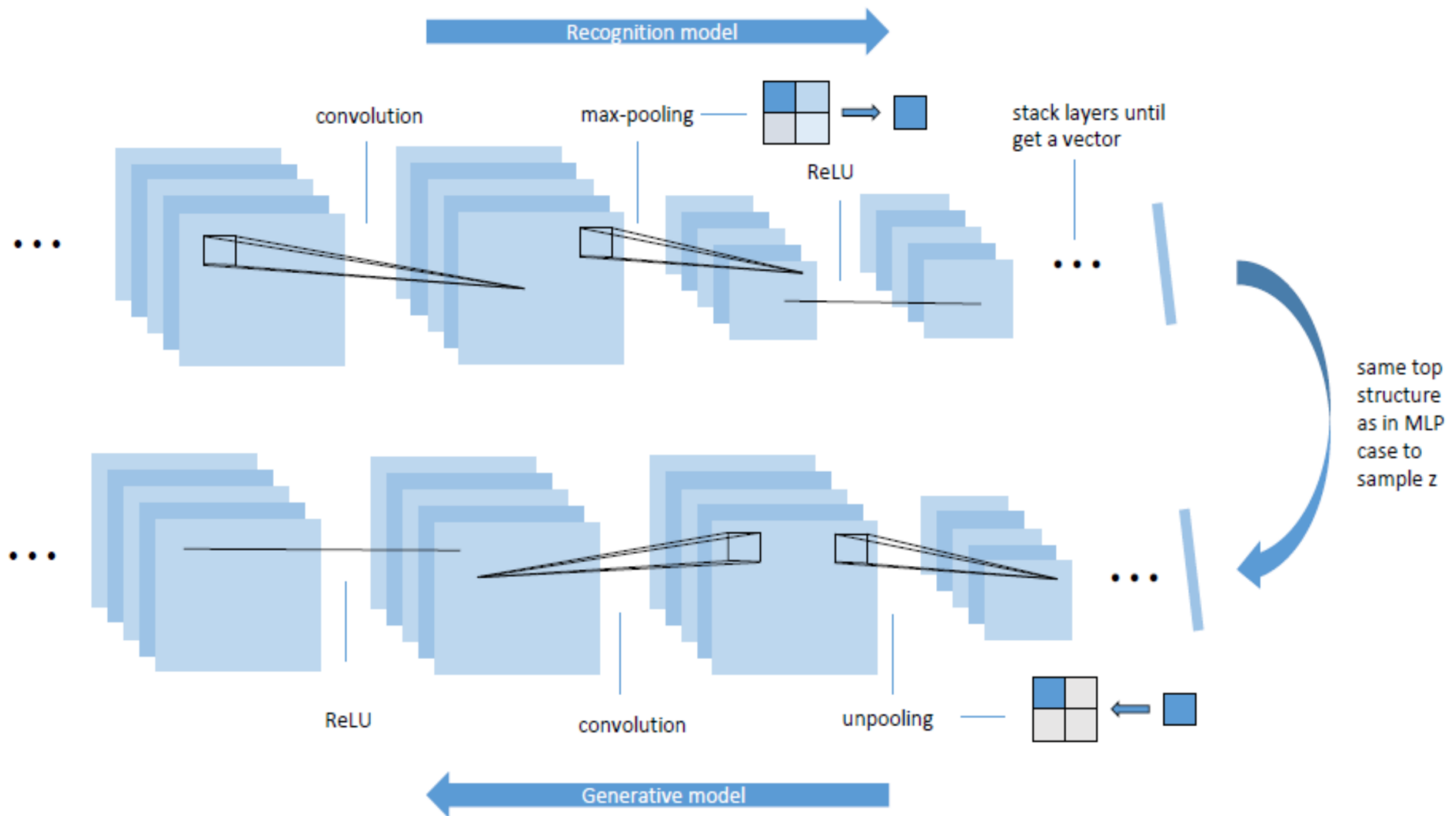


## 2-Layer MLP: Q-P network architecture



\*Same as in Auto-Encoding Variational Bayes (VA) [Kingma & Welling, 2014]

# 5-Layer CNN: Q-P network architecture



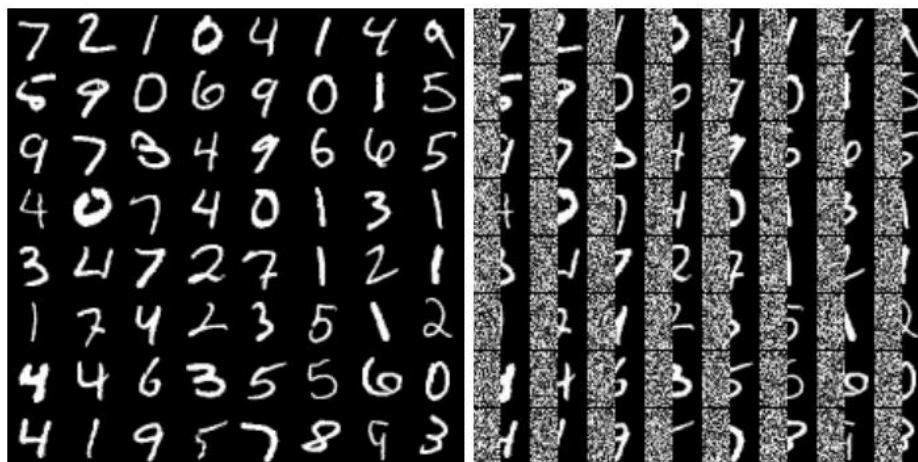
# Some Results

## ◆ Density estimation

MODELS	MNIST	OCR-LETTERS
<i>VAE</i>	-85.69	-30.09
<i>MEM-VAE(ours)</i>	<b>-84.41</b>	<b>-29.09</b>
<i>IWAE-5</i>	-84.43	-28.69
<i>MEM-IWAE-5(ours)</i>	<b>-83.26</b>	<b>-27.65</b>
<i>IWAE-50</i>	-83.58	-27.60
<i>MEM-IWAE-50(ours)</i>	<b>-82.84</b>	<b>-26.90</b>

- Better than symmetric VAE networks
- Comparable with state-of-the-art with much fewer parameters

# Missing Value Imputation



(a) Data

(b) Noisy data



(c) Results of VAE

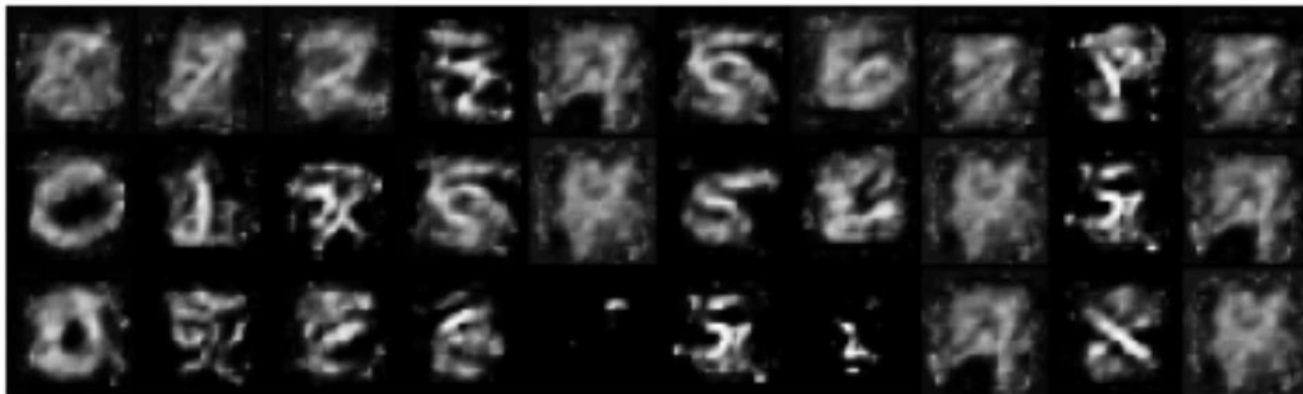
(d) Results of MEM-VAE

# Learnt Memory Slots

- ◆ Average preference over classes of the first 3 slots:

"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
0.27	0.82	0.33	0.11	0.34	0.15	0.49	0.27	0.09	0.28
0.24	0.09	0.06	0.11	0.30	0.13	0.12	0.27	0.09	0.21
0.18	0.05	0.06	0.11	0.07	0.07	0.05	0.11	0.09	0.18

- ◆ Corresponding images:



# References

- ◆ Kingma, D. P. and Welling, M. *Auto-encoding variational Bayes*. In ICLR, 2013.
- ◆ Rezende, D. J., Mohamed, S., and Wierstra, D. *Stochastic backpropagation and approximate inference in deep generative models*. In ICML, 2014.
- ◆ Burda, Y., Grosse, R., and Salakhutdinov, R. *Importance weighted autoencoders*. In arXiv:1509.00519, 2015.
- ◆ Goodfellow, I. J., Abadie, J. P., Mirza, M., Xu, B., Farley, D. W., S.ozair, Courville, A., and Bengio, Y. *Generative adversarial nets*. In NIPS, 2014
- ◆ Bengio, Y., Thlhodeau-Laufer, E., Alain, G., and Yosinski, J. *Deep generative stochastic networks trainable by backprop*. In ICML, 2014.
- ◆ Li, C., Zhu, J., and Zhang, B. *Learning to generate with memory*. In ICML, 2016

# Thanks!



**ZhuSuan: A Library for Bayesian Deep Learning.** [J. Shi](#), [J. Chen](#), [J. Zhu](#), [S. Sun](#), [Y. Luo](#), [Y. Gu](#), [Y. Zhou](#). arXiv preprint, arXiv:1709.05870 , 2017

**Online Documents:** <http://zhusuan.readthedocs.io/>