CRIPAC 智能感知与计算研究中心 Center for Research on Intelligent Perception and Computing

中国科学院自动化研究所 Institute of Automation Chinese Academy of Sciences

2018

Lecture 8: Approximate Inference

Wei Wang

Center for Research on Intelligent Perception and Computing (CRIPAC)

National Laboratory of Pattern Recognition (NLPR)

Institute of Automation, Chinese Academy of Science (CASIA)

Outline

1. Course Review
2. Markov Chain Monte Carlo
3. Variational Inference
4. Example: Deep Boltzmann Machines

Review: Generative Model

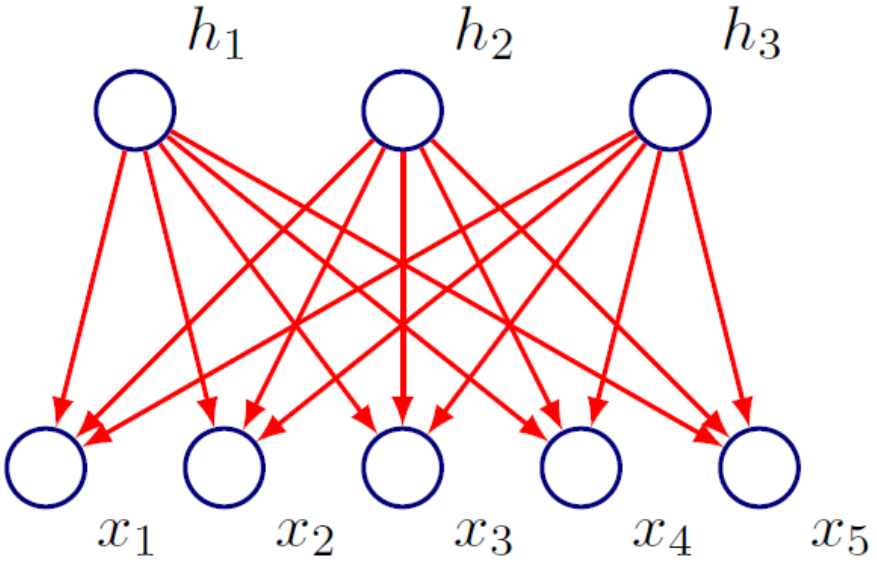
- We want to build a probabilistic model of the input $\tilde{P}(\mathbf{x})$
- Like before, we are interested in latent factors h that explain x
- We then care about the marginal:

$$\tilde{P}(\mathbf{x}) = \mathbb{E}_{\mathbf{h}} \tilde{P}(\mathbf{x}|\mathbf{h})$$

- h is a representation of the data

Review: Linear Factor Model

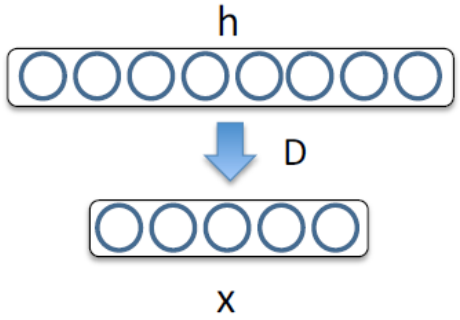
- The latent factor h is an encoding of the data
- Simplest decoding model: Get x after a linear transformation of x with some noise
- Formally: Suppose we sample the latent factors from a distribution $\mathbf{h} \sim P(\mathbf{h})$
- Then: x = Wh + b + ε



$$\mathbf{x} = W\mathbf{h} + \mathbf{b} + \epsilon$$

Review: Sparse Coding

- Sparse coding (Olshausen & Field, 1996). Originally developed to explain early visual processing in the brain (edge detection).
- For each input $\mathbf{x}^{(t)}$ find a latent representation $\mathbf{h}^{(t)}$ such that:
  - it is sparse: the vector $\mathbf{h}^{(t)}$ has many zeros
  - we can good reconstruct the original input $\mathbf{x}^{(t)}$



Review: Sparse Coding

- For each $\mathbf{x}^{(t)}$ find a latent representation $\mathbf{h}^{(t)}$ such that:
  - it is sparse: the vector $\mathbf{h}^{(t)}$ has many zeros
  - we can good reconstruct the original input $\mathbf{x}^{(t)}$
- In other words:

Reconstruction: $\widehat{\mathbf{x}}^{(t)}$

Sparsity vs. reconstruction control

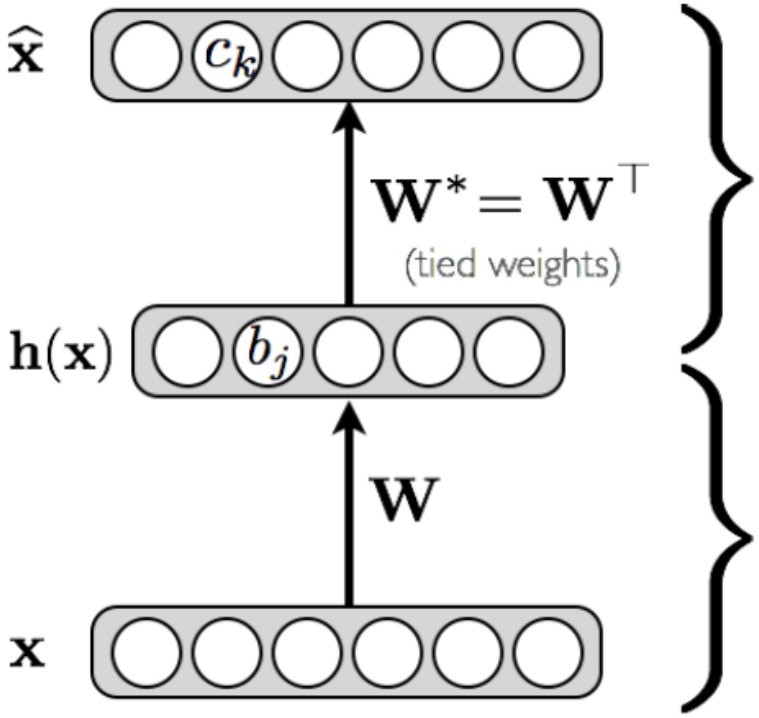
$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\,\mathbf{h}^{(t)}||_2^2 + \lambda ||\mathbf{h}^{(t)}||_1$$

Reconstruction error

Sparsity penalty

Review: Autoencoder

- Feed-forward neural network trained to reproduce its input at the output layer



Decoder

$$\hat{\mathbf{x}} = o(\hat{\mathbf{a}}(\mathbf{x})) = \mathrm{sigm}(\mathbf{c} + \mathbf{W}^*\mathbf{h}(\mathbf{x}))$$

For binary units

Encoder

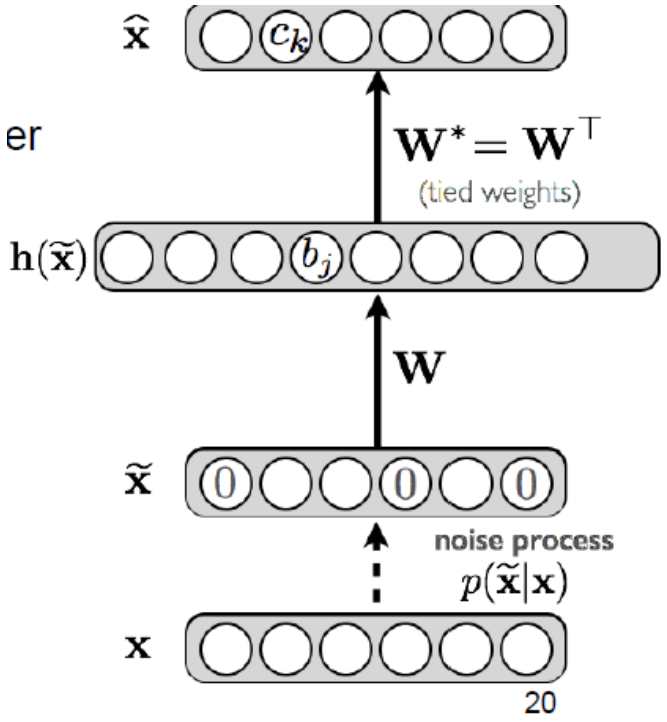
$$\mathbf{h}(\mathbf{x}) = g(\mathbf{a}(\mathbf{x})) = \mathrm{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x})$$

Review: Denoising Autoencoder

- Idea: representation should be robust to introduction of noise:
  - random assignment of subset of inputs to 0, with probability
  - Similar to dropouts on the input layer
  - Gaussian additive noise

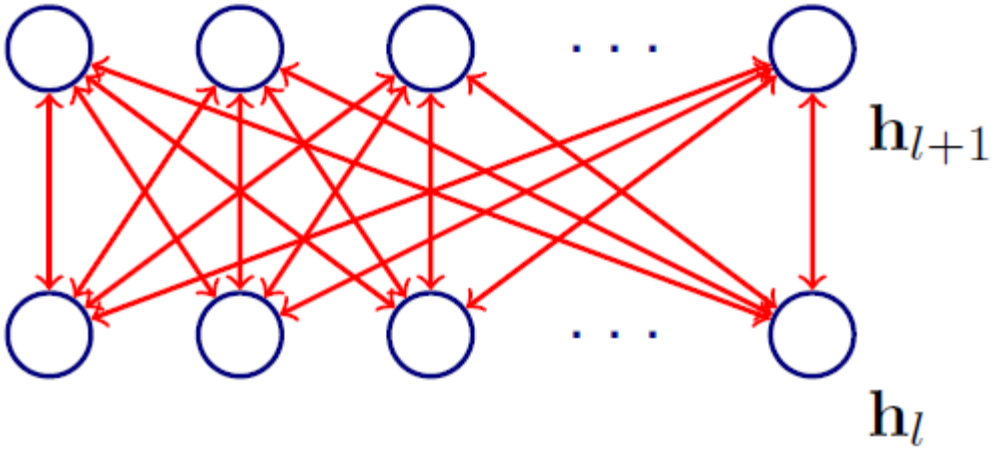
• Reconstruction $\widehat{\mathbf{x}}$ computed from the corrupted input $\widetilde{\mathbf{x}}$

• Loss function compares $\widehat{\mathbf{x}}$ reconstruction with the noiseless input $\mathbf{x}$



(Vincent et.al., ICML 2008)

Review: Restricted Boltzmann Machines



- x and h play symmetric roles:

$$P(\mathbf{x}|\mathbf{h}) = \prod_i P(\mathbf{x}_i|\mathbf{h})$$

- The common transfer (for the binary case):

$$P(\mathbf{h}_i = 1|\mathbf{x}) = \sigma(\mathbf{c_i} + W_i\mathbf{x})$$

$$P(\mathbf{x}_j = 1|\mathbf{h}) = \sigma(\mathbf{b_j} + W_{:,j}^T\mathbf{h})$$

Outline

1. Course Review
2. Markov Chain Monte Carlo
3. Variational Inference
4. Example: Deep Boltzmann Machines

Approximate Inference

- When using probabilistic graphical models, we will be interested in evaluating the posterior distribution $p(\mathbf{Z}|\mathbf{X})$ of the latent variables $\mathbf{Z}$ given the observed data $\mathbf{X}$.

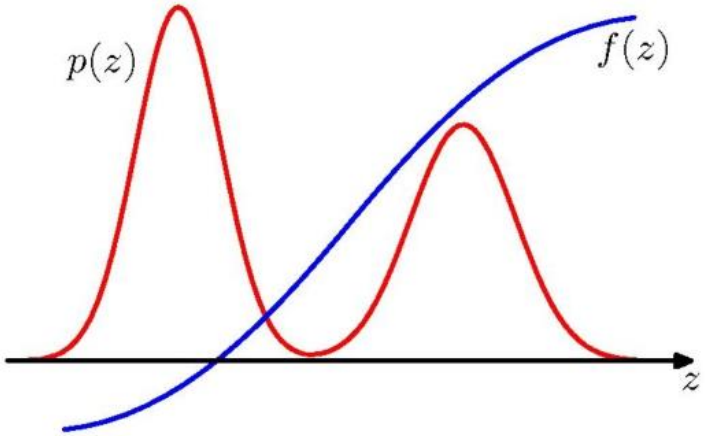
- For example, in the EM algorithm, we need to evaluate the expectation of the complete-data log-likelihood with respect to the posterior distribution over the latent variables.

- For more complex models, it may be infeasible to evaluate the posterior distribution, or compute expectations with respect to this distribution.

- We now consider sampling-based methods, known as Monte Carlo techniques.

Notation

- For most situations, we will be interested in evaluating expectations (for example in order to make predictions):



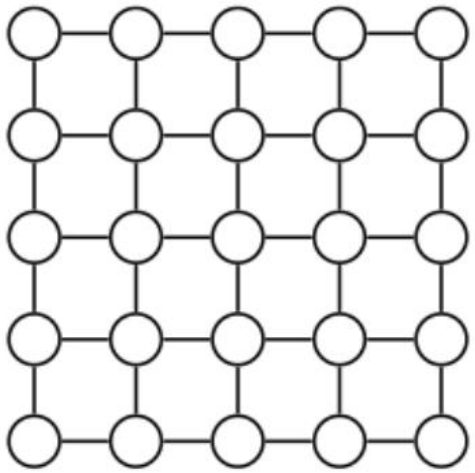
$$\mathbb{E}[f] = \int f(\mathbf{z})p(\mathbf{z})\mathrm{d}\mathbf{z}.$$

where the integral will be replaced with summation in case of discrete variables.

- We will make use of the following notation: $p(\mathbf{z}) = \frac{\tilde{p}(\mathbf{z})}{\mathcal{Z}}.$
- We can evaluate $\tilde{p}(\mathbf{z})$ pointwise but cannot evaluate $\mathcal{Z}$.
  - Posterior distribution: $p(\theta|\mathcal{D}) = \frac{1}{p(\mathcal{D})}p(\mathcal{D}|\theta)p(\theta).$
  - Markov Random Fields: $p(\mathbf{x}) = \frac{1}{\mathcal{Z}}\exp(-E(\mathbf{x})).$

Undirected Graphical Models

- Let **x** be a binary random vector with $x_i \in \{-1.1\}$:



$$P_\theta(\mathbf{x}) = \frac{1}{\mathcal{Z}(\theta)} \exp\Big(\sum_{ij \in E} x_i x_j \theta_{ij} + \sum_{i \in V} x_i \theta_i \Big)$$

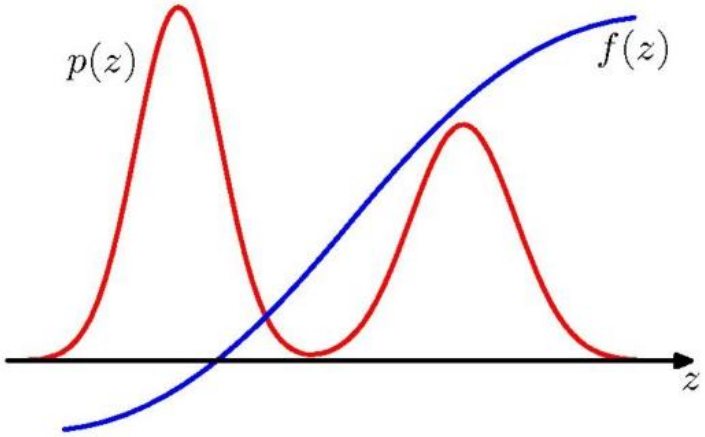
where Z(μ) is a normalizing constant (also known as partition function):

$$\mathcal{Z}(\theta) = \sum_{\mathbf{x}} \exp\Big(\sum_{ij \in E} x_i x_j \theta_{ij} + \sum_{i \in V} x_i \theta_i \Big).$$

- If **x** is 100-dimensional, we need to sum over 2^{100} terms.
- The sum might decompose, which would be the case for the tree structured graphical models (or models with low tree-width). Otherwise, we need to approximate.

Simple Monte Carlo

- General Idea: Draw independent samples $\{z_1,..,z_n\}$ from distribution p(z) to approximate expectation:



$$\mathbb{E}[f] = \int f(z)p(z)dz \approx \frac{1}{N}\sum_{n=1}^{N} f(z^n) = \hat{f}.$$

Note that:

$$\mathbb{E}[f] = \mathbb{E}[\hat{f}].$$

so the estimator has correct mean (unbiased).

- The variance:

$$\mathrm{var}[\hat{f}] = \frac{1}{N}\mathbb{E}\left[(f - \mathbb{E}[f])^2\right].$$

- Variance decreases as 1/N.
- **Remark**: The accuracy of the estimator does not depend on dimensionality of z.

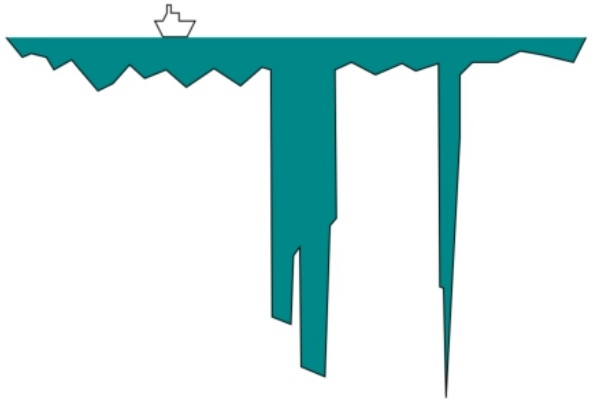
Simple Monte Carlo

- In general:

$$\mathbb{E}[f] = \int f(z)p(z)dz \approx \frac{1}{N}\sum_{n=1}^{N} f(z^n), \quad z^n \sim p(z).$$

- Predictive distribution:

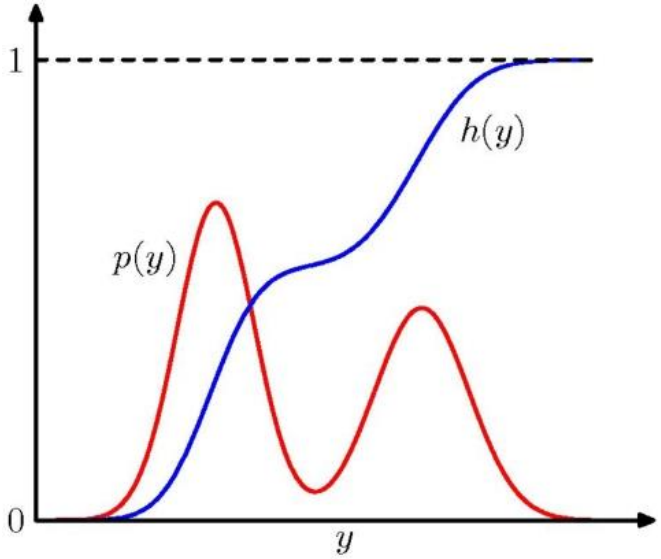
$$p(x^*|\mathcal{D}) = \int p(x^*|\theta,\mathcal{D})p(\theta|\mathcal{D})d\theta$$
$$\approx \frac{1}{N}\sum_{n=1}^{N} p(x^*|\theta^n), \quad \theta^n \sim p(\theta|\mathcal{D}).$$



- **Problem**: It is hard to draw exact samples from p(z).

Basic Sampling Algorithm

• How can we generate samples from simple non-uniform distributions assuming we can generate samples from uniform distribution.



• Define:

$$h(y) = \int_{-\infty}^{y} p(\hat{y})d\hat{y}.$$

• Sample:

$$z \sim \mathrm{U}[0, 1]$$

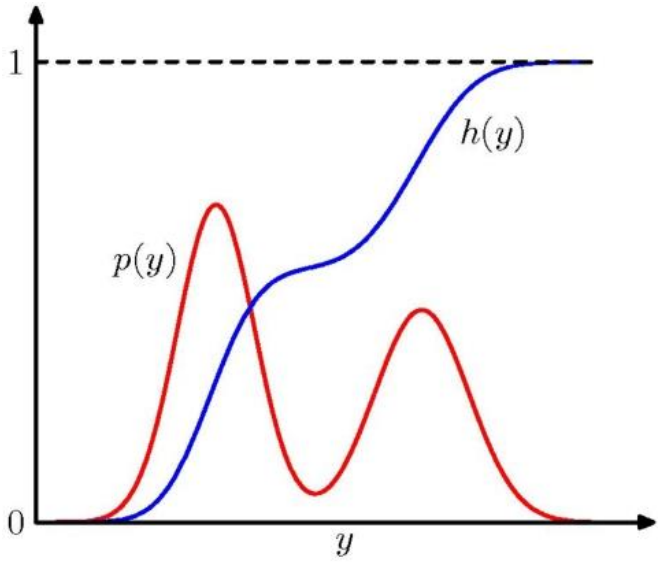
• Then $y = h^{-1}(z)$

is a sample from p(y).

Basic Sampling Algorithm

- For example, consider the exponential distribution:

$$p(y) = \lambda \exp(-\lambda y).$$



- In this case:

$$h(y) = \int_0^y p(\hat{y}) d\hat{y} = 1 - \exp(-\lambda y).$$

- Sample:

$$z \sim \mathrm{U}[0, 1]$$

- Then

$$y = h^{-1}(z) = -\lambda^{-1} \ln(1 - z)$$

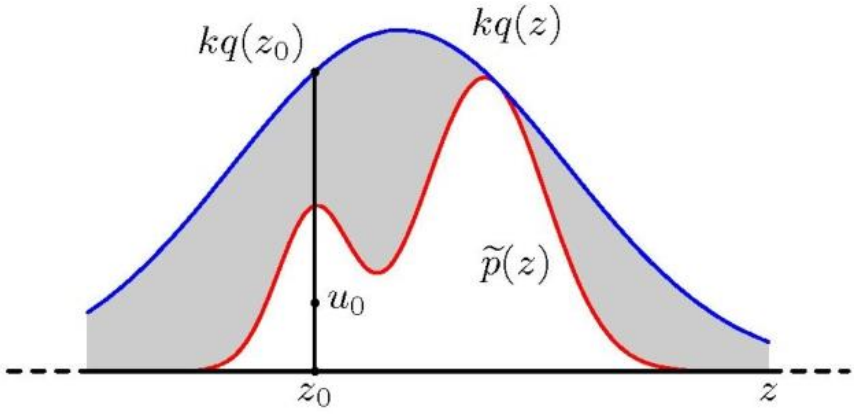
is a sample from p(y).

- **Problem**: Computing h(y) is just as hard!

Rejection Sampling

- Sampling from the target distribution $\tilde{p}(z)$ is difficult. Suppose we have an easy-to-sample proposal distribution q(z), such that:

$$kq(z) \geq \tilde{p}(z), \ \forall z.$$



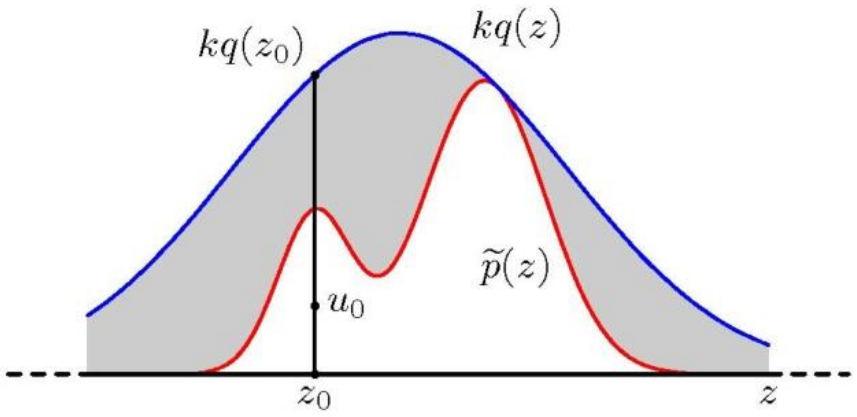
- Sample:
  $z_0 \sim q(z)$,
- Sample:
  $u_0 \sim \text{Uniform}[0, kq(z_0)]$.

- Sample (z_0, u_0) has uniform distribution under the curve of $kq(z)$.
- If $u_0 > \tilde{p}(z_0)$, the sample is rejected.

Rejection Sampling

- Probability that a sample is accepted is calculated as:

$$p(\text{accept}) = \int \frac{\tilde{p}(z)}{kq(z)} q(z)dz$$
$$= \frac{1}{k} \int \tilde{p}(z)dz.$$

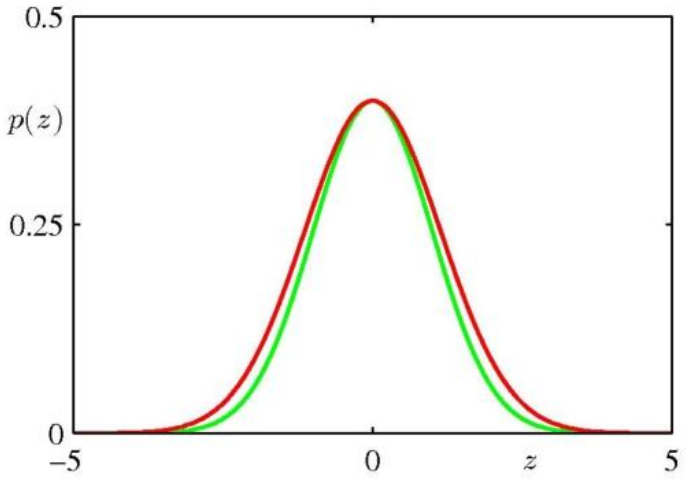


- The fraction of accepted samples depends on the ratio of the area under $\tilde{p}(z)$ and $kq(z)$.

- It is often hard to find q(z) with optimal k.

Rejection Sampling

- Consider the following simple problem:



- Target distribution:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|0, \sigma_p^2 I),$$

- Proposal distribution:

$$q(\mathbf{z}) = \mathcal{N}(\mathbf{z}|0, \sigma_q^2 I).$$

- We must have:

$$\sigma_q^2 \geq \sigma_p^2.$$

- The optimal k is given by: $k = \left(\frac{\sigma_q}{\sigma_p}\right)^D$.

- Hence the acceptance rate diminishes exponentially!

- Useful technique in one or two dimensions. Typically applies as a subroutine in more advanced techniques.

Importance Sampling

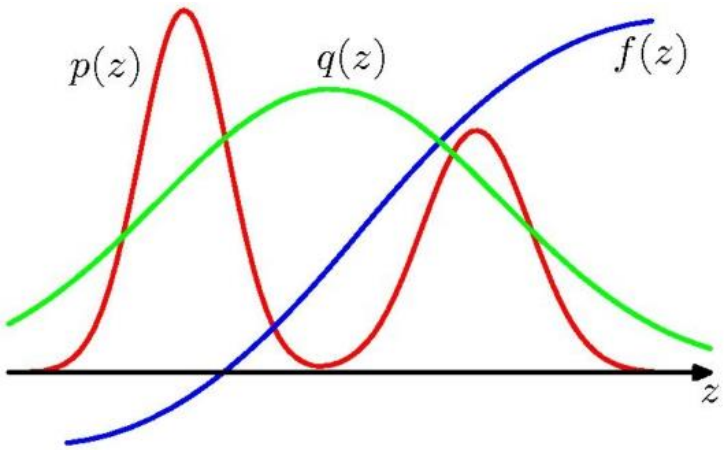
- Suppose we have an easy-to-sample proposal distribution q(z), such that

$$q(z) > 0 \text{ if } p(z) > 0.$$

$$\mathbb{E}[f] = \int f(z)p(z)dz$$

$$= \int f(z)\frac{p(z)}{q(z)}q(z)dz$$

$$\approx \frac{1}{N}\sum_n \frac{p(z^n)}{q(z^n)}f(z^n), \; z^n \sim q(z).$$



- The quantities

$$w^n = p(z^n)/q(z^n)$$

are known as **importance weights**.

- Unlike rejection sampling all samples are retained.
- But wait: we cannot compute $p(z) = \frac{\tilde{p}(z)}{Z}$.

Importance Sampling

- Let our proposal be of the form: $q(z) = \tilde{q}(z)/\mathcal{Z}_q$.

$$\mathbb{E}[f] = \int f(z)p(z)dz = \int f(z)\frac{p(z)}{q(z)}q(z)dz = \frac{Z_q}{Z_p}\int f(z)\frac{\tilde{p}(z)}{\tilde{q}(z)}q(z)dz$$
$$\approx \frac{\mathcal{Z}_q}{\mathcal{Z}_p}\frac{1}{N}\sum_n \frac{\tilde{p}(z^n)}{\tilde{q}(z^n)}f(z^n) = \frac{\mathcal{Z}_q}{\mathcal{Z}_p}\frac{1}{N}\sum_n w^n f(z^n),$$

- But we can use the same weights to approximate $\mathcal{Z}_q/\mathcal{Z}_p$:

$$\frac{\mathcal{Z}_p}{\mathcal{Z}_q} = \frac{1}{\mathcal{Z}_q}\int \tilde{p}(z)dz = \int \frac{\tilde{p}(z)}{\tilde{q}(z)}q(z)dz \approx \frac{1}{N}\sum_n \frac{\tilde{p}(z^n)}{\tilde{q}^n(z)} = \frac{1}{N}\sum_n w^n.$$

- Hence:

$$\mathbb{E}[f] \approx \sum_{n=1}^{N} \frac{w^n}{\sum_{m=1}^{N} w^m} f(z^n), \qquad z^n \sim q(z).$$

Summary so Far

- If our proposal distribution q(z) poorly matches our target distribution p(z) then:
  - Rejection sampling: almost always rejects
  - Importance Sampling: has large, possibly infinite, variance (unreliable estimator).
- For high-dimensional problems, finding good proposal distributions is very hard.

What can we do?

- Markov Chain Monte Carlo.

Markov Chains

- A first-order Markov chain: a series of random variables $\{z^1, ..., z^N\}$, such that the following conditional independence property holds for $n \in \{z^1, ..., z^{N-1}\}$:

$$p(z^{n+1}|z^1, .., z^n) = p(z^{n+1}|z^n).$$

- We can specify Markov chain:
  - Probability distribution for initial state $p(z^1)$.
  - Conditional probability for subsequent states in the form of transition probabilities:

$$T(z^{n+1} \leftarrow z^n) = p(z^{n+1}|z^n).$$

- $T(z^{n+1} \leftarrow z^n)$ is often called a **transition kernel**.

Markov Chains

• A marginal probability of a particular state can be computed as:

$$p(z^{n+1}) = \sum_{z^n} T(z^{n+1} \leftarrow z^n) p(z^n).$$

• A distribution $\pi(z)$ is said to be **invariant** or **stationary** with respect to a Markov chain if each step in the chain leaves $\pi(z)$ invariant:

$$\pi(z) = \sum_{z'} T(z \leftarrow z') \pi(z').$$

• A given Markov chain may have many stationary distributions.

• For example:

$$T(z \leftarrow z') = I(z = z')$$

is the identity transformation. Then any distribution is invariant.

Detailed Balance

• A sufficient (but not necessary) condition for ensuring that ¼(z) is invariant is to choose a transition kernel that satisfies a **detailed balance property**:

$$\pi(z')T(z \leftarrow z') = \pi(z)T(z' \leftarrow z).$$

$$T(z' \leftarrow z) = p(z'|z).$$

• A transition kernel that satisfies detailed balance will leave that distribution invariant:

$$\sum_{z'} \pi(z')T(z \leftarrow z') = \sum_{z'} \pi(z)T(z' \leftarrow z)$$

$$= \pi(z) \sum_{z'} T(z' \leftarrow z) = \pi(z).$$

• A Markov chain that satisfies detailed balance is said to be **reversible**.

Example

- Discrete example:

$$P^\star = \begin{pmatrix} 3/5 \\ 1/5 \\ 1/5 \end{pmatrix} \qquad T = \begin{pmatrix} 2/3 & 1/2 & 1/2 \\ 1/6 & 0 & 1/2 \\ 1/6 & 1/2 & 0 \end{pmatrix} \qquad T_{ij} = T(x_i \leftarrow x_j)$$

- In this case P^* is **invariant distribution** of T since $TP^* = P^*$, or:

$$\sum_{z'} P^*(z')T(z \leftarrow z') = P^*(z).$$

- P^* is also the **equilibrium distribution** of T since:

$$T^{100} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 3/5 \\ 1/5 \\ 1/5 \end{pmatrix} = P^\star$$

Markov Chains

- We want to sample from the target distribution (e.g. posterior distribution, or a Markov Random Field): $\pi(z) = \tilde{\pi}(z)/\mathcal{Z}.$
- Obtaining independent samples is difficult.
  - Set up a Markov chain with transition kernel T(z' ← z) that leaves our target distribution $\pi(z)$ invariant.
  - If the chain is ergodic, then the chain will converge to this unique equilibrium distribution.
  - We obtain dependent samples drawn approximately from $\pi(z)$ by simulating a Markov chain for some time.
- **Ergodicity** requires: There exists K, such that for any starting z, we have

$$T^K(z' \leftarrow z) > 0 \text{ for all } \pi(z') > 0.$$

Combining Transition Operators

- In practice, we often construct the transition probabilities from a set of "**base**" transition operators $B_1,\ldots,B_K$.
- One option is to consider a mixture distribution of the form:

$$T(z' \leftarrow z) = \sum_{k=1}^{K} \alpha_k B_k(z' \leftarrow z),$$

where mixing coefficients satisfy: $\alpha_k \geq 0, \quad \sum_k \alpha_k = 1.$

- Another option is to combine through successive application:

$$T(z' \leftarrow z) = \sum_{z^1} \ldots \sum_{z^{n-1}} B_1(z' \leftarrow z^1) \ldots B_K(z^{K-1} \leftarrow z).$$

Combining Transition Operators

- For the case of the mixture:

$$T(z' \leftarrow z) = \sum_{k=1}^{K} \alpha_k B_k(z' \leftarrow z),$$

If each of the base distributions satisfies the detailed balance, then the mixture transition T will also satisfy detailed balance.

- For the case of using composite transition probabilities:

$$T(z' \leftarrow z) = \sum_{z^1} ... \sum_{z^{n-1}} B_1(z' \leftarrow z^1) ... B_K(z^{K-1} \leftarrow z).$$

this does not hold.

- A simple idea is to **symmetrize** the order of application of the base transitions:

$$B_1, B_2, ..., B_K, B_K, ..., B_2, B_1.$$

- A common example of using composite transition probabilities is where each base transition changes only a subset of variables.

Metropolis-Hasting Algorithm

• A Markov chain transition operator from the current state z to a new state z' is defined as follows:

- A new "candidate" state z^* is proposed according to some proposal distribution $q(z^*|z)$.

- A candidate z^* is accepted with probability:

$$\min\left(1, \frac{\tilde{\pi}(z^*)}{\tilde{\pi}(z)} \frac{q(z|z^*)}{q(z^*|z)}\right).$$

- If accepted, set z' = z^*. Otherwise z'= z, or the next state is the copy of the current state.

• Note: there is no need to compute normalizing constant.

• For symmetric proposals, the acceptance probability reduces to:

$$\min\left(1, \frac{\tilde{\pi}(z^*)}{\tilde{\pi}(z)}\right).$$

Metropolis-Hasting Algorithm

- We can show that M-H transition kernel leaves $\pi(z)$ invariant by showing that it satisfies detailed balance:

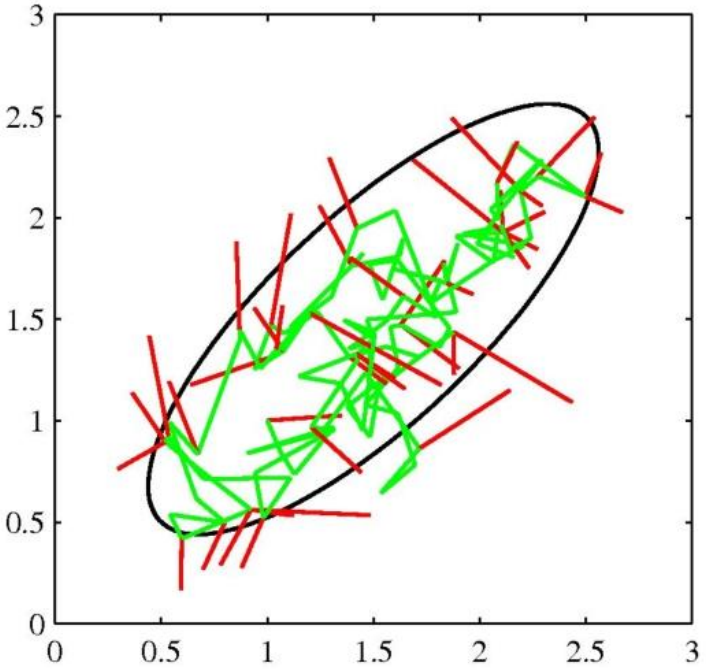
$$\begin{aligned}
\pi(z)T(z' \leftarrow z) &= \pi(z)q(z'|z)\min\left(1, \frac{\pi(z')}{\pi(z)}\frac{q(z|z')}{q(z'|z)}\right) \\
&= \min\left(\pi(z)q(z'|z), \pi(z')q(z|z')\right) \\
&= \pi(z')q(z|z')\min\left(\frac{\pi(z)}{\pi(z')}\frac{q(z'|z)}{q(z|z')}, 1\right) \\
&= \pi(z')T(z \leftarrow z').
\end{aligned}$$

- Note that whether the chain is ergodic will depend on the particulars of the stationary distribution $\pi(z)$ and proposal distribution q.

Metropolis-Hasting Algorithm

- Using Metropolis algorithm to sample from Gaussian distribution with proposal

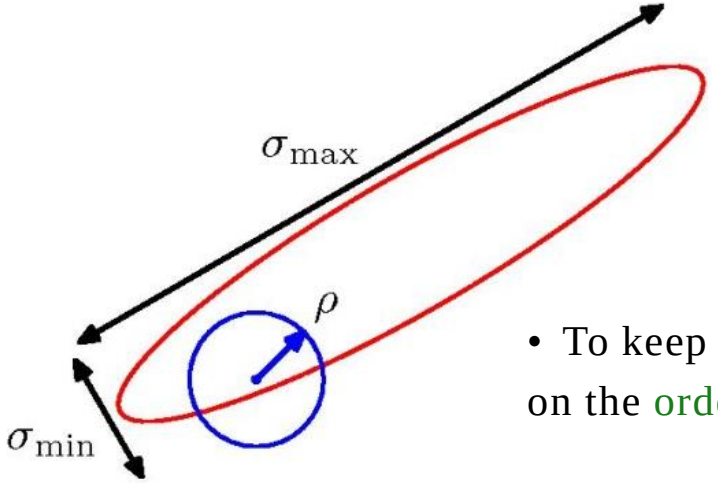
$$q(z'|z) = \mathcal{N}(z, 0.04).$$



- accepted (green), rejected (red).
- 150 samples were generated and 43 were rejected.
- Note that generated samples are not independent.

Choice of Proposal

- Suppose that our goal is to sample from the correlated multivariate Gaussian distribution.
- Consider a Gaussian proposal: centered on the current state:



$$q(z'|z) = \mathcal{N}(z, \rho^2 I)$$

- ρ large -- many rejections
- ρ small -- chain moves too slowly.

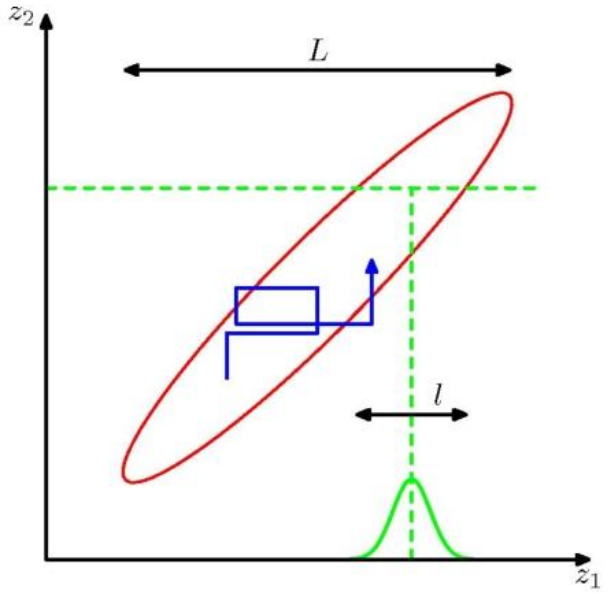
- To keep the rejection rate low, the scale ρ should be on the order of the smallest standard deviation σ_{min}.

- Random walk behaviour: The number of steps separating states that are approximately independent is of order: $(\sigma_{max}/\sigma_{min})^2$.

- The specific choice of proposal can greatly affect the performance of the algorithm.

Gibbs Sampler

- Consider sampling from $p(z_1,\ldots,z_N)$:



- Initialize z_i, i=1,..,N.

- For t=1:T

- Sample: $z_1^{t+1} \sim p(z_1|z_2^t,...,z_N^t)$
- Sample: $z_2^{t+1} \sim p(z_2|z_1^{t+1},z_3^t,...,z_N^t)$
- …
- Sample: $z_N^{t+1} \sim p(z_N|z_1^{t+1},z_2^{t+1},...,z_{N-1}^{t+1})$

- This procedure samples from the required distribution p(z).

- When sampling $p(z_n|\mathbf{z}_{-n})$ the marginal distribution $p(\mathbf{z}_{-n})$ is clearly invariant, as it does not change.
- Each step samples from the correct conditional, hence the joint distribution is itself invariant.

Gibbs Sampler

• Applicability of the Gibbs sampler depends on how easy it is to sample from conditional probabilities $p(z_n|\mathbf{z}_{-n})$.

• For discrete random variables with a few discrete settings:

$$p(z_n|\mathbf{z}_{-n}) = \frac{p(z_n, \mathbf{z}_{-n})}{\sum_{z_n} p(z_n, \mathbf{z}_{-n})},$$

where the sum can be performed analytically.

• For continuous random variables:

$$p(z_n|\mathbf{z}_{-n}) = \frac{p(z_n, \mathbf{z}_{-n})}{\int p(z_n, \mathbf{z}_{-n}) dz_n},$$

• The integral is univariate and is often analytically tractable or amenable to standard sampling methods.

Gibbs Sampler

- Gibbs sampler is a particular instance of M-H algorithm with proposals: $q_n(\mathbf{z}^*|\mathbf{z}) = p(z_n^*|\mathbf{z}_{-n})$.

- Note that $\mathbf{z}_{-n}^* = \mathbf{z}_{-n}$ because these components are unchanged by the sampling step.

- Let us look at the factor that determines acceptance probability in M-H.

$$A(\mathbf{z}^*, \mathbf{z}) = \frac{p(\mathbf{z}^*)}{p(\mathbf{z})} \times \frac{q_n(\mathbf{z}|\mathbf{z}^*)}{q_n(\mathbf{z}^*|\mathbf{z})}$$
$$= \frac{p(z_n^*|\mathbf{z}_{-n}^*)p(\mathbf{z}_{-n}^*)}{p(z_n|\mathbf{z}_{-n})p(\mathbf{z}_{-n})} \times \frac{p(z_n|\mathbf{z}_{-n}^*)}{p(z_n^*|\mathbf{z}_{-n})} = 1.$$

- Thus MH steps are always accepted.

- Let us look at the behavior of Gibbs.

Auxiliary Variables

- The goal of MCMC is to marginalize out variables.
- But sometimes it is useful to introduce additional, or auxiliary variables.

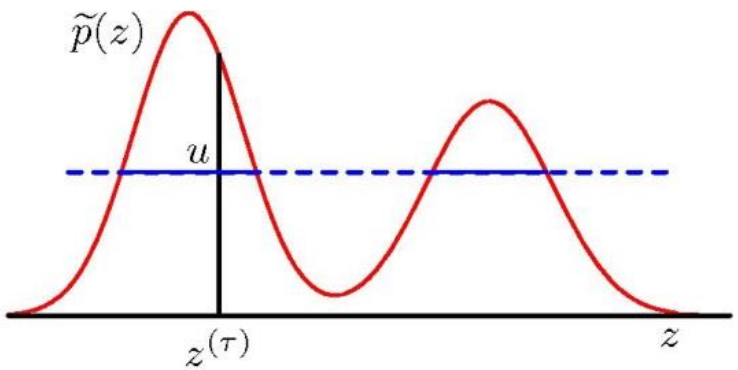
$$\int f(z)p(z)dz = \int f(z)p(z,u)dzdu$$

$$\approx \frac{1}{L}\sum_{l=1}^{L} f(z^l), \qquad (z,u) \sim p(z,u).$$

- We would want to do this if:
  - Sampling from conditionals p(z | u) and p(u | z) is easy.
  - It is easier to deal with p(z,u).
- Many MCMC algorithms use this idea.

Slice Sampling

- M-H algorithm is sensitive to the step size.
- Slice sampling provides an adaptive step size that is automatically adjusted.
- We augment z with an additional (auxiliary) variable u and then draw samples from the joint (z,u) space.



(a)

- The goal is to sample uniformly from the area under the distribution:

$$\hat{p}(z,u) = \begin{cases} 1/\mathcal{Z}_p & 0 \leq u \leq \tilde{p}(z) \\ 0 & \text{otherwise} \end{cases}$$

- The marginal distribution over z is:

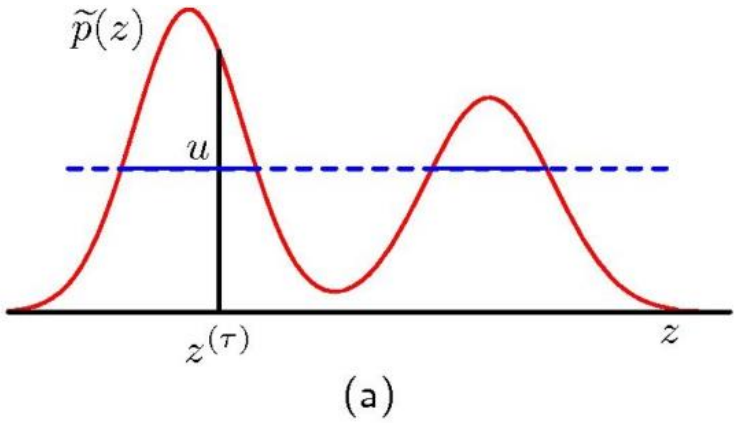
$$\int \tilde{p}(z,u)\mathrm{d}u = \int_0^{\tilde{p}(z)} \frac{1}{\mathcal{Z}_p}\mathrm{d}u = \frac{\tilde{p}(z)}{\mathcal{Z}_p} = p(z),$$

which is the target distribution of interest.

Slice Sampling

- The goal if sample uniformly from the area under the distribution:

$$\hat{p}(z,u) = \begin{cases} 1/\mathcal{Z}_p & 0 \le u \le \tilde{p}(z) \\ 0 & \text{otherwise} \end{cases}$$



(a)

- Given z, we sample u uniformly from:

$$0 \le u \le \tilde{p}(z),$$

which is easy.

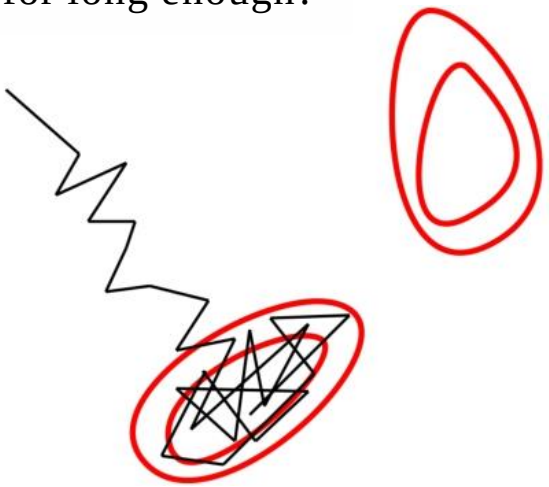
- Given u, we sample z uniformly from the **slice** through the distribution defined:

$$\{z : \tilde{p}(z) > u\}.$$

- In practice, sampling directly from a slice might be difficult.
- Instead we can define a sampling scheme that leaves the distribution $\hat{p}(z,u)$ invariant.

Using MCMC in Practice

- The samples we obtain from MCMC are not independent. Should we think, i.e. only keep every Kth sample?
- We often start MCMC from arbitrary starting points. Should be discard a burn-in period?
- Should we perform multiple runs as opposed to one long run?
- How do we know whether we have run our chain for long enough?



Outline

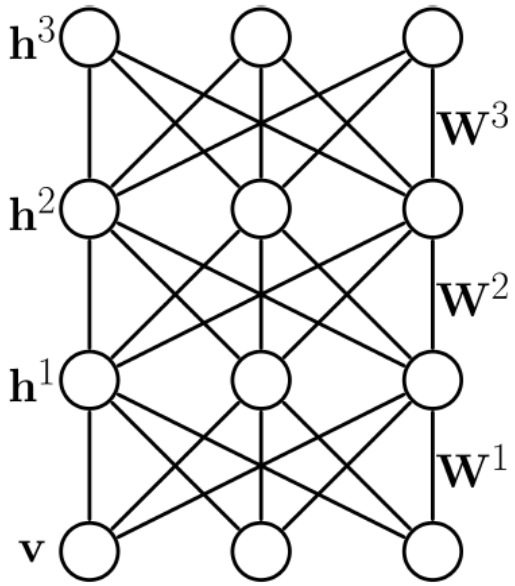
1. Course Review
2. Markov Chain Monte Carlo
3. Variational Inference
4. Example: Deep Boltzmann Machines

Approximate Inference

- When using probabilistic graphical models, we will be interested in evaluating the posterior distribution $p(Z|X)$ of the latent variables Z given the observed data X.
- For example, in the EM algorithm, we need to evaluate the expectation of the complete-data log-likelihood with respect to the posterior distribution over the latent variables.
- For more complex models, it may be infeasible to evaluate the posterior distribution, or compute expectations with respect to this distribution.
- This typically occurs when working with high-dimensional latent spaces, or when the posterior distribution has a complex form, for which expectations are not analytically tractable (e.g. Boltzmann machines).

Deep Boltzmann Machines

For many interesting real-world problems, we need to introduce hidden or latent variables.



- Our random variables will contain both visible and hidden variables x=(v,h).

$$p(\mathbf{v}) = \frac{1}{\mathcal{Z}} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}))$$

- In general, computing, computing posterior over hidden variables will be intractable:

$$P_{data}(\mathbf{v}, \mathbf{h^1}) = P_\theta(\mathbf{h^1}|\mathbf{v}) P_{data}(\mathbf{v})$$

$$P_{data}(\mathbf{v}) = \frac{1}{N} \sum_{n=1}^{N} \delta(\mathbf{v} - \mathbf{v_n})$$

Variational Bound

- Given a joint distribution p(Z,X|θ) over observed and latent variables governed by parametersθ, the goal is to maximize the likelihood function p(X|θ) with respect toθ:

$$p(\mathbf{X}|\theta) = \sum_{Z} p(\mathbf{X}, \mathbf{Z}|\theta).$$

- We will assume that Z is discrete, although derivations are identical if Z contains continuous, or a combination of discrete and continuous variables.
- For any distribution q(Z) over latent variables we can derive the following variational lower bound:

$$\ln p(\mathbf{X}|\theta) = \ln \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\theta) = \ln \sum_{\mathbf{Z}} q(\mathbf{Z}) \frac{p(\mathbf{X}, \mathbf{Z}|\theta)}{q(\mathbf{Z})}$$

Jensen's inequality

$$\geq \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z}|\theta)}{q(\mathbf{Z})} = \mathcal{L}(q, \theta).$$

Variational Bound

- Variational lower-bound:

$$\ln p(\mathbf{X}|\theta) = \ln \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\theta) = \ln \sum_{\mathbf{Z}} q(\mathbf{Z}) \frac{p(\mathbf{X}, \mathbf{Z}|\theta)}{q(\mathbf{Z})}$$

$$\geq \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z}|\theta)}{q(\mathbf{Z})}$$

$$= \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln p(\mathbf{X}, \mathbf{Z}|\theta) + \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{1}{q(\mathbf{Z})}$$

$$= \mathbb{E}_{q(\mathbf{Z})} \left[\ln p(\mathbf{X}, \mathbf{Z}|\theta) \right] + \mathcal{H}(q(\mathbf{Z})) = \mathcal{L}(q, \theta).$$

Expected complete log-likelihood

Entropy functional.

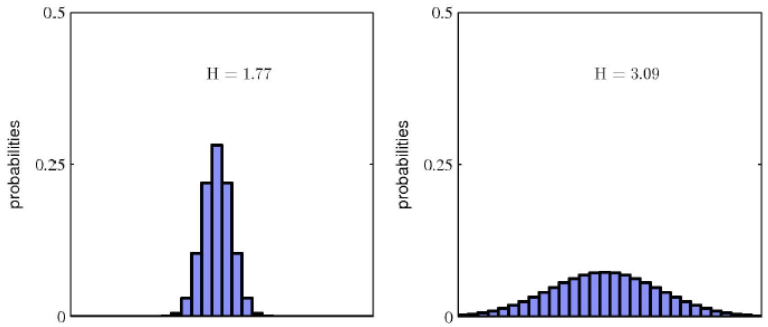
Variational lower-bound

Entropy

- For a discrete random variable X, where $P(X=x_i) = p(x_i)$, the entropy of a random variable is:

$$\mathcal{H}(p) = -\sum_i p(x_i) \log p(x_i).$$

- Distributions that are sharply picked around a few values will have a relatively low entropy, whereas those that are spread more evenly across many values will have higher entropy



- Histograms of two probability distributions over 30 bins.
- The largest entropy will arise from a uniform distribution $H = -\ln(1/30) = 3.40$.

- For a density defined over continuous random variable, the differential entropy is given by: $\mathcal{H}(p) = -\int p(x) \log p(x) \mathrm{d}x.$

Variational Bound

- We saw:

$$\ln p(\mathbf{X}|\theta) \geq \mathbb{E}_{q(\mathbf{Z})}\left[\ln p(\mathbf{X},\mathbf{Z}|\theta)\right] + \mathcal{H}(q(\mathbf{Z})) = \mathcal{L}(q,\theta).$$

- We also note that the following decomposition holds:

Where $$\ln p(\mathbf{X}|\theta) = \mathcal{L}(q,\theta) + \mathrm{KL}(q||p),$$

Variational lower-bound

$$\mathcal{L}(q,\theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z}|\theta)}{q(\mathbf{Z})},$$

Kullback-Leibler (KL) divergence.

Also known as Relative Entropy.

$$\mathrm{KL}(q||p) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X},\theta)}{q(\mathbf{Z})}.$$

- KL divergence is not symmetric.
- KL(q||p) ¸ 0 with equality iff p(x) = q(x)
- Intuitively, it measures the “distance” between the two distributions.

Variational Bound

- Let us derive that:

$$\log p(\mathbf{X}|\theta) = \mathcal{L}(q,\theta) + \mathrm{KL}(q||p),$$

- We can write:

$$\ln p(\mathbf{X},\mathbf{Z}|\theta) = \ln p(\mathbf{Z}|\mathbf{X},\theta) + \ln p(\mathbf{X}|\theta),$$

and plugging into the definition of $\mathcal{L}(q,\theta)$, gives the desired result.

- Note that variational bound becomes tight iff q(Z) = p(Z |X,θ).
- In other words the distribution q(Z) is equal to the true posterior distribution over the latent variables, so that KL(q||p) = 0.
- As KL(q||p) $\geq$0, it immediately follows that:

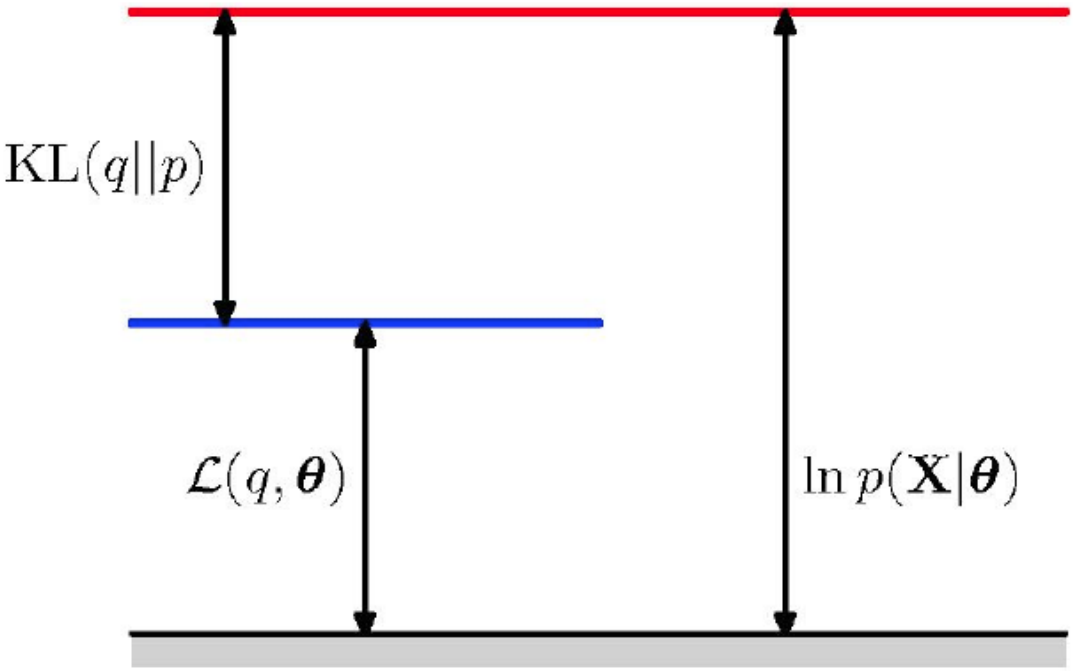
$$\ln p(\mathbf{X}|\theta) \geq \mathcal{L}(q,\theta),$$

which also showed using Jensen's inequality.

Decomposition

- Illustration of the decomposition which holds for any distribution q(Z).

$$\ln p(\mathbf{X}|\theta) = \mathcal{L}(q,\theta) + \mathrm{KL}(q||p),$$



Variational Bound

- We can decompose the marginal log-probability as:

$$\log p(\mathbf{X}) = \mathcal{L}(q) + \mathrm{KL}(q||p),$$

where

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \mathrm{d}\mathbf{Z}$$

$$\mathrm{KL}(q||p) = -\int q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \mathrm{d}\mathbf{Z}.$$

- We can maximize the variational lower bound $\mathcal{L}(q)$ with respect to the distribution q(Z), which is equivalent to minimizing the KL divergence.
- If we allow any possible choice of q(Z), then the maximum of the lower bound occurs when: $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X})$.

  In this case KL divergence becomes zero.

Variational Bound

- As in our previous lecture, we can decompose the marginal log-probability as:

$$\log p(\mathbf{X}) = \mathcal{L}(q) + \mathbf{KL}(q||p),$$

- We will assume that the true posterior distribution is intractable.
- We can consider a restricted family of distributions q(Z) and then find the member of this family for which KL is minimized.
- Our goal is to restrict the family of distributions so that it contains only tractable distributions.
- At the same time, we want to allow the family to be sufficiently rich and flexible, so that it can provide a good approximation to the posterior.
- One option is to use parametric distributions $q(\mathbf{Z}|\omega)$, governed by parameters ω.
- The lower bound then becomes a function of ω. and we can optimize the lowerbound to determine the optimal values for the parameters.

Mean-Field

- We now consider restricting the family of distributions.
- Partition the elements of Z into M disjoint groups, denoted by Z_i, i=1,…,M.
- We assume that the q distribution factorizes with respect to these groups:

$$q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}_i).$$

- Note that we place no restrictions on the functional form of the individual factors q_i (we will often denote $q_i(Z_i)$ as simply q_i).
- This approximation framework, developed in physics, is called mean-field theory.

Factorized Distributions

- Among all factorized distributions, we look for a distribution for which the variational lower bound is maximized.
- Denoting $q_i(Z_i)$ as simply q_i, we have:

$$q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}_i).$$

$$\begin{aligned}
\mathcal{L}(q) &= \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \mathrm{d}\mathbf{Z} \\
&= \int \prod_i q_i \left[\ln p(\mathbf{X}, \mathbf{Z}) - \sum_i \ln q_i \right] \mathrm{d}\mathbf{Z} \\
&= \int q_j \left[\int \ln p(\mathbf{X}, \mathbf{Z}) \prod_{i \neq j} q_i \, \mathrm{d}\mathbf{Z}_i \right] \mathrm{d}\mathbf{Z}_j - \int q_j \ln q_j \, \mathrm{d}\mathbf{Z}_j + \text{const} \\
&= \int q_j \ln \tilde{p}(\mathbf{X}, \mathbf{Z}_j) \, \mathrm{d}\mathbf{Z}_j - \int q_j \ln q_j \, \mathrm{d}\mathbf{Z}_j + \text{const}
\end{aligned}$$

where we denote a new distribution:

$$\tilde{p}(\mathbf{X}, \mathbf{Z}_j) = \mathbb{E}_{i \neq j} [\ln p(\mathbf{X}, \mathbf{Z})] + \text{const.}$$

Factorized Distributions

- Among all factorized distributions, we look for a distribution for which the variational lower bound is maximized.
- Denoting $q_i(\boldsymbol{Z}_i)$ as simply q_i, we have:

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \mathrm{d}\mathbf{Z}$$
$$= \int q_j \ln \tilde{p}(\mathbf{X}, \mathbf{Z}_j) \, \mathrm{d}\mathbf{Z}_j - \int q_j \ln q_j \, \mathrm{d}\mathbf{Z}_j + \text{const}$$

where

$$\ln \tilde{p}(\mathbf{X}, \mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const.}$$

- Here we take an expectation with respect to the q distribution over all variables $\boldsymbol{Z}_i$ for $i \neq j$, so that:

$$\mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] = \int \ln p(\mathbf{X}, \mathbf{Z}) \prod_{i \neq j} q_i \, \mathrm{d}\mathbf{Z}_i.$$

Maximizing Lower Bound

- Now suppose that we keep $\{q_{i\neq j}\}$ fixed, and optimize the lower bound with respect to all possible forms of the distribution $q_i(\boldsymbol{Z}_i)$.
- This optimization is easily done by recognizing that:

$$\mathcal{L}(q) = \int q_j \ln \tilde{p}(\mathbf{X}, \mathbf{Z}_j)\, d\mathbf{Z}_j - \int q_j \ln q_j\, d\mathbf{Z}_j + \text{const}$$

$$= -\text{KL}(q_j(\mathbf{Z}_j)||\tilde{p}(\mathbf{X}, \mathbf{Z}_j) + \text{const},$$

constant: does not depend on q.

$$\mathcal{L}(q) = \log p(\mathbf{X}) - \text{KL}(q||p)$$

so the minimum occurs when

$$q_j^*(\mathbf{Z}_j) = \tilde{p}(\mathbf{X}, \mathbf{Z}), \text{ or } \quad \ln q_j^*(\mathbf{Z}_j) = \mathbb{E}_{i\neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const.}$$

- Observe: the log of the optimum solution for factor q_i is given by:
  - Considering the log of the joint distribution over all hidden and visible variables
  - Taking the expectation with respect to all other factors $\{q_i\}$ for $i \neq j$.

Maximizing Lower Bound

- Exponentiating and normalizing, we obtain:

$$q_j^*(\mathbf{Z}_j) = \frac{\exp\left(\mathbb{E}_{i\neq j}\left[\ln p(\mathbf{X},\mathbf{Z})\right]\right)}{\int \exp\left(\mathbb{E}_{i\neq j}\left[\ln p(\mathbf{X},\mathbf{Z})\right]\right)\mathrm{d}\mathbf{Z}_j}.$$

- The set of these equations for $j = 1, \ldots, M$ represent the set of consistency conditions for the maximum of the lower bound subject to factorization constraint.
- To obtain a solution, we initialize all of the factors and then cycle through factors, replacing each in tern with a revised estimate.
- Convergence is guaranteed because the bound is convex with respect to each of the individual factors.

Factorized Gaussian

- Consider a problem of approximating a general distribution by a factorized distribution.
- To get some insight, let us look at the problem of approximating a Gaussian distribution using a factorized Gaussian distribution.
- Consider a Gaussian distribution over two correlated variables $\boldsymbol{z} = (z_1, z_2)$.

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}),$$

$$\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \qquad \boldsymbol{\Lambda} = \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{12} & \beta_{22}. \end{pmatrix}$$

- Let us approximate this distribution using a factorized Gaussian of the form:

$$q(\mathbf{z}) = q_1(z_1)q_2(z_2).$$

Factorized Gaussian

- Remember:

$$\ln q_j^*(\mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const.}$$

- Consider an expression for the optimal factor q_1:

$$\begin{aligned}
\ln q_1^*(z_1) &= \mathbb{E}_{q_2(z_2)}[\ln p(\mathbf{z})] + \text{const} \\
&= \mathbb{E}_{q_2(z_2)}\left[-\frac{\beta_{11}}{2}(z_1 - \mu_1)^2 - \beta_{12}(z_1 - \mu_1)(z_2 - \mu_2) \right] + \text{const} \\
&= -\frac{\beta_{11}}{2} z_1^2 + \beta_{11} z_1 \mu_1 - \beta_{12} z_1 (\mathbb{E}[z_2] - \mu_2) + \text{const.}
\end{aligned}$$

- Note that we have a quadratic function of z_1, and so we can identify $q_1(z_1)$ as a Gaussian distribution:

$$q_1^*(z_1) = \mathcal{N}(z_1 | m_1, \beta_{11}^{-1}), \quad m_1 = \mu_1 - \frac{\beta_{12}}{\beta_{11}}(\mathbb{E}[z_2] - \mu_2).$$

Factorized Gaussian

- By symmetry, we also obtain:

$$q_1^*(z_1) = \mathcal{N}(z_1|m_1, \beta_{11}^{-1}), \quad m_1 = \mu_1 - \frac{\beta_{12}}{\beta_{11}}(\mathbb{E}[z_2] - \mu_2).$$

$$q_2^*(z_2) = \mathcal{N}(z_2|m_2, \beta_{22}^{-1}), \quad m_2 = \mu_2 - \frac{\beta_{12}}{\beta_{22}}(\mathbb{E}[z_1] - \mu_1).$$

- There are two observations to make:
  - We did not assume that $q_i^*(z_i)$ is Gaussian, but rather we derived this result by optimizing variational bound over all possible distributions.
  - The solutions are coupled. The optimal $q_1^*(z_1)$ depends on expectation computed with respect to $q_2^*(z_2)$.
- One option is to cycle through the variables in turn and update them until convergence.

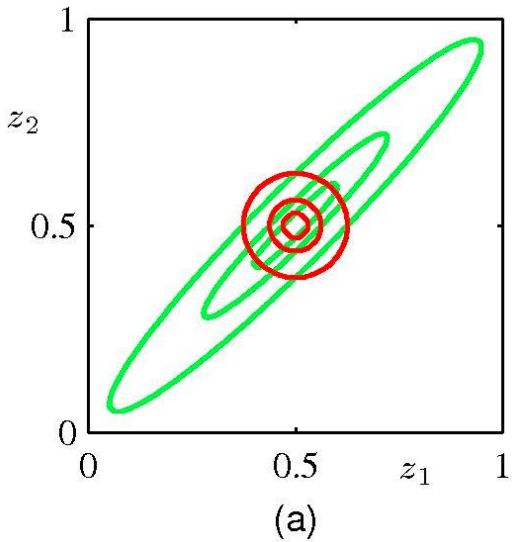
Factorized Gaussian

- By symmetry, we also obtain:

$$q_1^*(z_1) = \mathcal{N}(z_1|m_1, \beta_{11}^{-1}), \quad m_1 = \mu_1 - \frac{\beta_{12}}{\beta_{11}}(\mathbb{E}[z_2] - \mu_2).$$

$$q_2^*(z_2) = \mathcal{N}(z_2|m_2, \beta_{22}^{-1}), \quad m_2 = \mu_2 - \frac{\beta_{12}}{\beta_{22}}(\mathbb{E}[z_1] - \mu_1).$$

- However, in our case, $\mathbb{E}[z_1] = \mu_1, \ \mathbb{E}[z_2] = \mu_2.$
- The green contours correspond to 1,2, and 3 standard deviations of the correlated Gaussian.
- The red contours correspond to the factorial approximation q(**z**) over the same two variables.
- Observe that a factorized variational approximation tends to give approximations that are too compact.



Alternative Form of KL Divergence

- We have looked at the variational approximation that minimizes KL($q||p$).
- For comparison, suppose that we were minimizing KL($p||q$).

$$\mathrm{KL}(p||q) = -\int p(\mathbf{Z}) \ln \frac{q(\mathbf{Z})}{p(\mathbf{Z})} \mathrm{d}\mathbf{Z}.$$

$$\mathrm{KL}(p||q) = -\int p(\mathbf{Z}) \left[\sum_{i=1}^{M} \ln q_i(\mathbf{Z}_i) \right] \mathrm{d}\mathbf{Z} + \int p(\mathbf{Z}) \ln \frac{1}{p(\mathbf{Z})} \mathrm{d}\mathbf{Z}.$$

constant: does not depend on q.

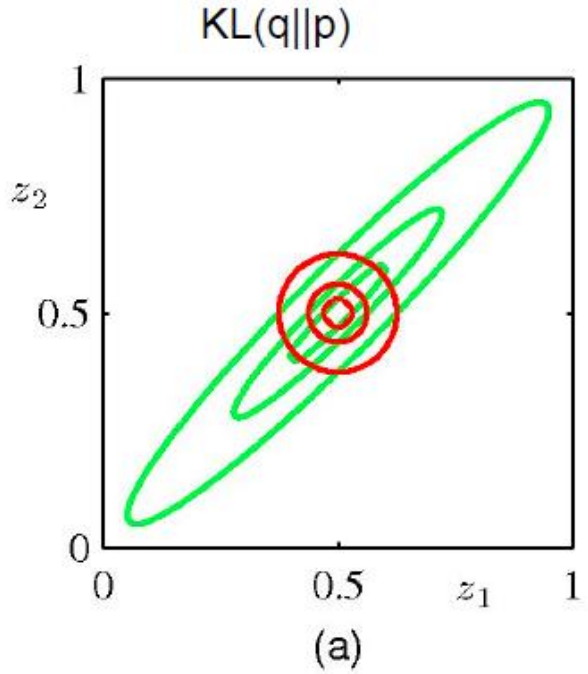
- It is easy to show that:

$$q_j^*(\mathbf{Z}_j) = \int p(\mathbf{Z}) \prod_{i \neq j} d\mathbf{Z}_i = p(\mathbf{Z}_j).$$

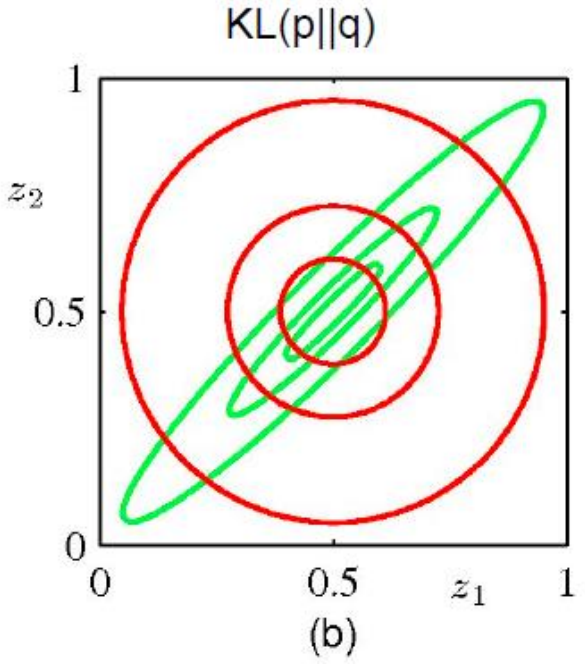
- The optimal factor is given by the marginal distribution of p(**Z**).

Comparison of two KLs

- Comparison of two the alternative forms for the KL divergence.



Approximation is too compact.

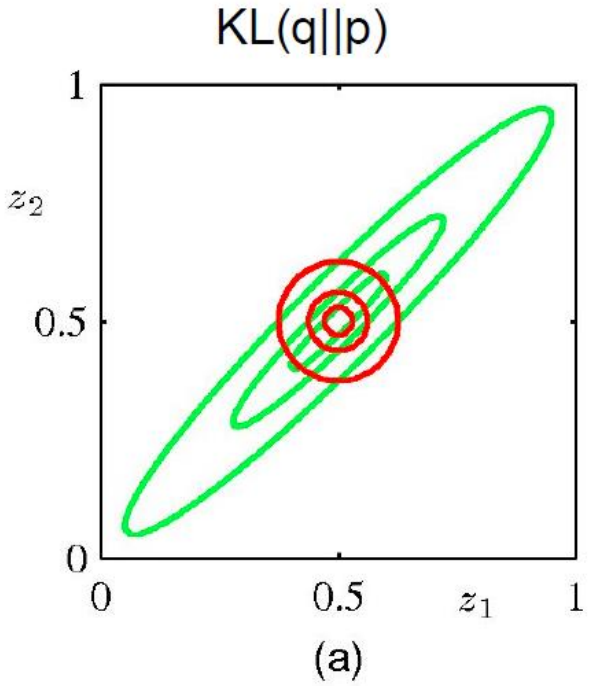


Approximation is too spread.

Comparison of two KLs

- The difference between these two approximations can be understood as follows:

$$\mathrm{KL}(q||p) = -\int q(\mathbf{Z}) \ln \frac{p(\mathbf{Z})}{q(\mathbf{Z})} \mathrm{d}\mathbf{Z}$$



- There is a large positive contribution to the KL divergence from regions of **Z** space in which:
  - p(**Z**) is near zero,
  - unless q(**Z**) is also close to zero.

- Minimizing KL($q||p$) leads to distributions q(**Z**) that avoid regions in which p(**Z**) is small.

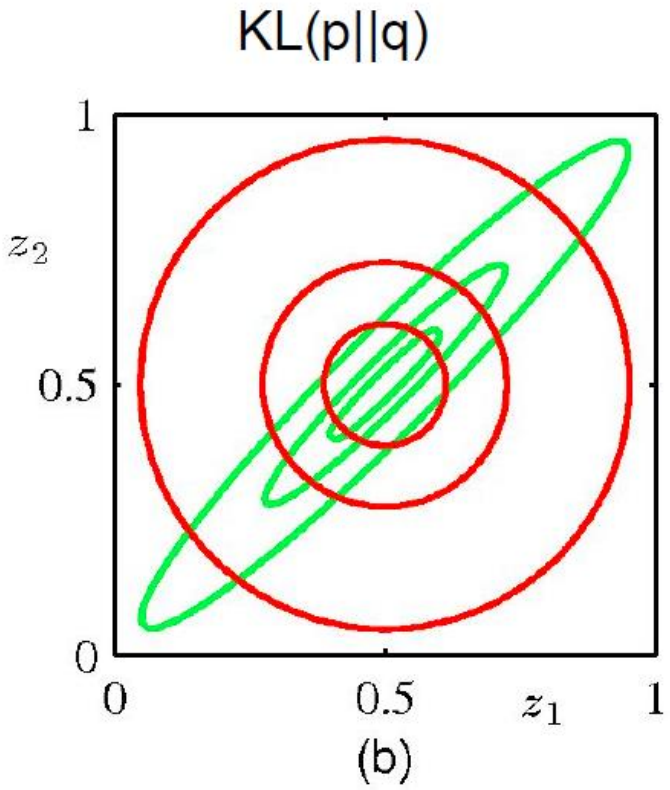
Comparison of two KLs

- Similar arguments apply for the alternative KL divergence:

$$\mathrm{KL}(p||q) = -\int p(\mathbf{Z}) \ln \frac{q(\mathbf{Z})}{p(\mathbf{Z})} \mathrm{d}\mathbf{Z}.$$

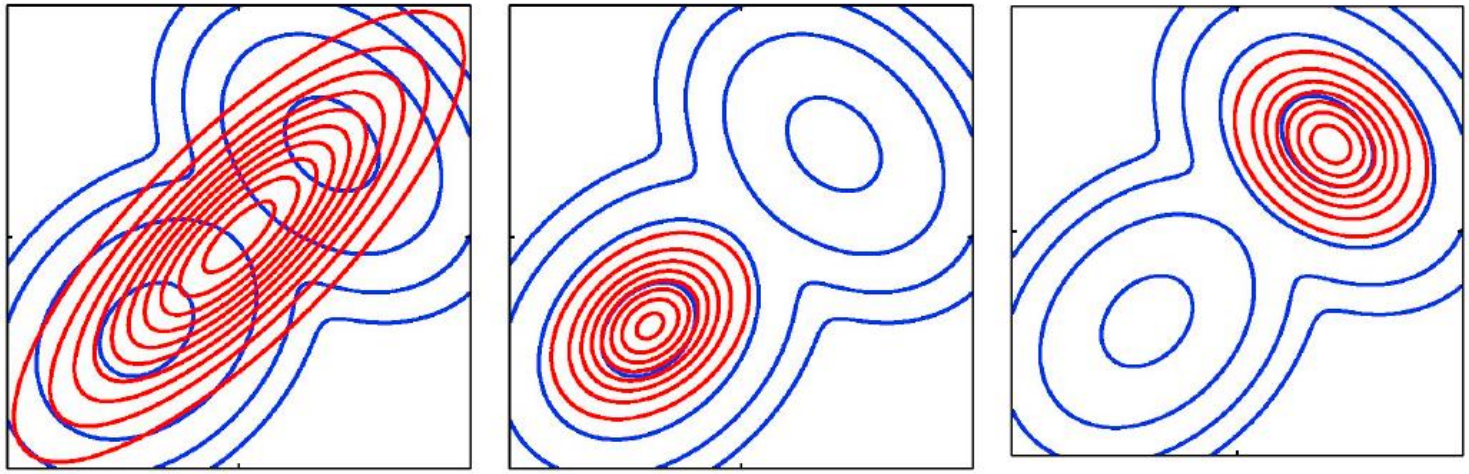
- There is a large positive contribution t the KL divergence from regions of **Z** space in which:
  - q(**Z**) is near zero,
  - unless p(**Z**) is also close to zero.

- Minimizing KL($p||q$) leads to distributions q(**Z**) that are nonzero in regions where p(**Z**) is nonzero.



Approximating Multimodal Distribution

- Consider approximating multimodal distribution with a unimodal one.
- Blue contours show bimodal distribution p(**Z**), red contours show a single Gaussian distribution that best approximates q(**Z**) that best approximates p(**Z**).



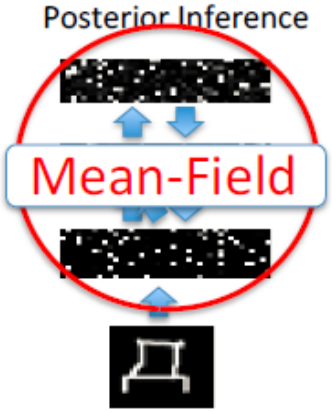
KL(p||q) KL(q||p) KL(q||p)

- In practice, the true posterior will often be mutlimodal.
- KL($q||p$) will tend to find a single mode, whereas KL($p||q$) will average across all of the modes.

Variational Inference

Approximate intractable distribution $P_\theta(\mathbf{h}|\mathbf{v})$ with simpler, tractable distribution $Q_\mu(\mathbf{h}|\mathbf{v})$:

$$\log P_\theta(\mathbf{v}) = \log \sum_{\mathbf{h}} P_\theta(\mathbf{h}, \mathbf{v}) = \log \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \frac{P_\theta(\mathbf{h}, \mathbf{v})}{Q_\mu(\mathbf{h}|\mathbf{v})}$$



$$\geq \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \log \frac{P_\theta(\mathbf{h}, \mathbf{v})}{Q_\mu(\mathbf{h}|\mathbf{v})}$$

$$= \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \underbrace{\log P^*_\theta(\mathbf{h}, \mathbf{v})}_{\mathbf{v}^\top W^1 \mathbf{h}^1 + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3} - \log \mathcal{Z}(\theta) + \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \log \frac{1}{Q_\mu(\mathbf{h}|\mathbf{v})}$$

Variational Lower Bound

$$= \log P_\theta(\mathbf{v}) - \mathrm{KL}\big(Q_\mu(\mathbf{h}|\mathbf{v}) || P_\theta(\mathbf{h}|\mathbf{v})\big)$$

$$\mathrm{KL}(Q||P) = \int Q(x) \log \frac{Q(x)}{P(x)} dx$$

Minimize KL between approximating and true distributions with respect to variational parameters μ.

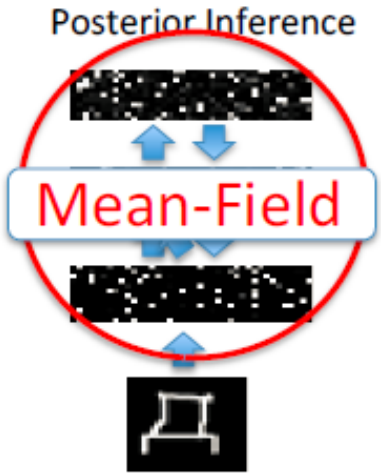
(Salakhutdinov, 2008; Salakhutdinov & Larochelle, AI & Statistics 2010)

Variational Inference

Approximate intractable distribution $P_\theta(\mathbf{h}|\mathbf{v})$ with simpler, tractable distribution $Q_\mu(\mathbf{h}|\mathbf{v})$:

$$\mathrm{KL}(Q||P) = \int Q(x) \log \frac{Q(x)}{P(x)} dx$$

$$\log P_\theta(\mathbf{v}) \geq \underbrace{\log P_\theta(\mathbf{v}) - \mathrm{KL}\big(Q_\mu(\mathbf{h}|\mathbf{v})||P_\theta(\mathbf{h}|\mathbf{v})\big)}_{\text{Variational Lower Bound}}$$



Mean-Field: Choose a fully factorized distribution:

$$Q_\mu(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{F} q(h_j|\mathbf{v}) \text{ with } q(h_j = 1|\mathbf{v}) = \mu_j$$

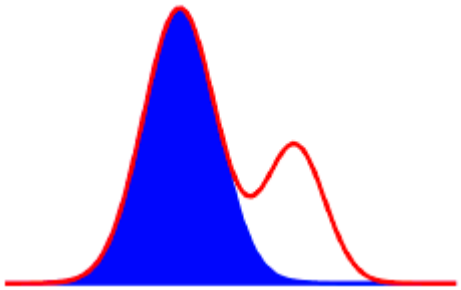
Variational Inference: Maximize the lower bound w.r.t. Variational parameters μ.

Nonlinear fixed-point equations:

$$\mu_j^{(1)} = \sigma\Big(\sum_i W_{ij}^1 v_i + \sum_k W_{jk}^2 \mu_k^{(2)}\Big)$$

$$\mu_k^{(2)} = \sigma\Big(\sum_j W_{jk}^2 \mu_j^{(1)} + \sum_m W_{km}^3 \mu_m^{(3)}\Big)$$

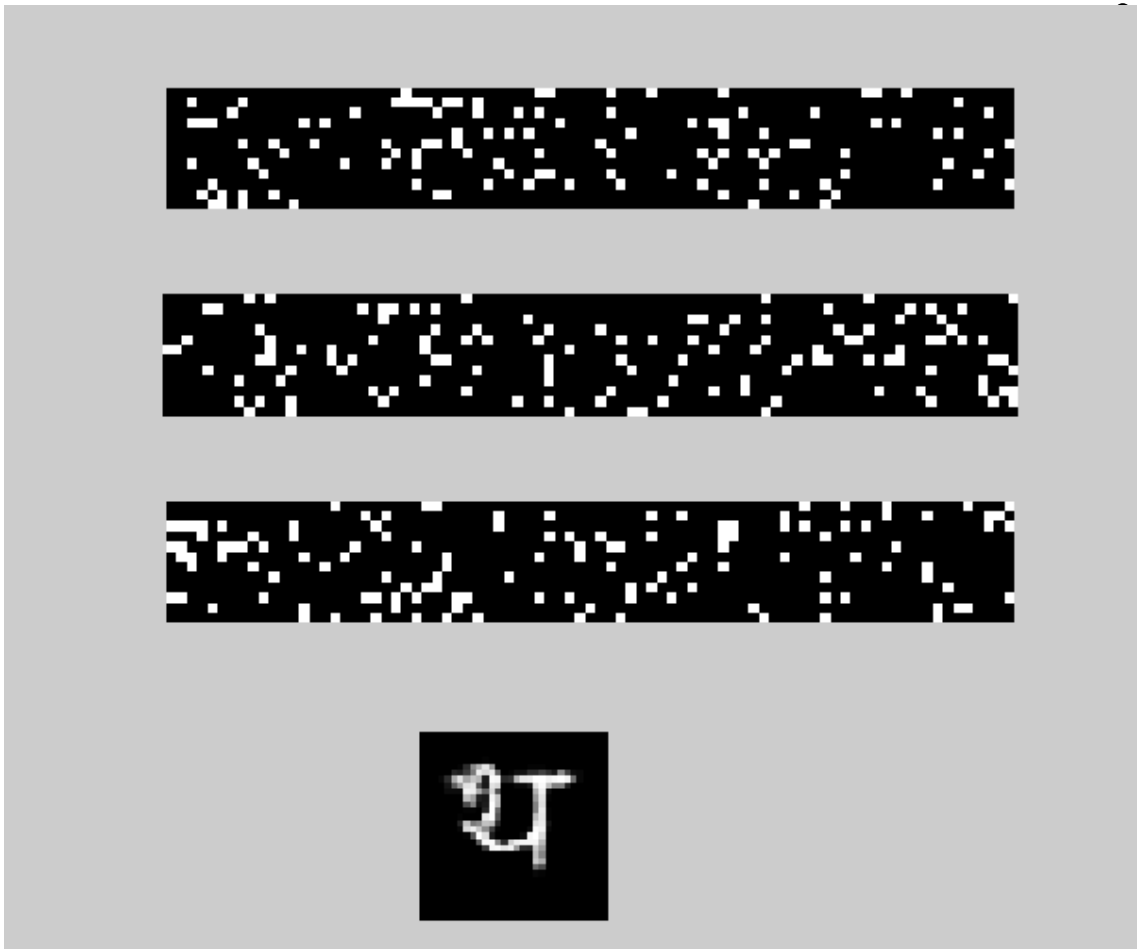
$$\mu_m^{(3)} = \sigma\Big(\sum_k W_{km}^3 \mu_k^{(2)}\Big)$$



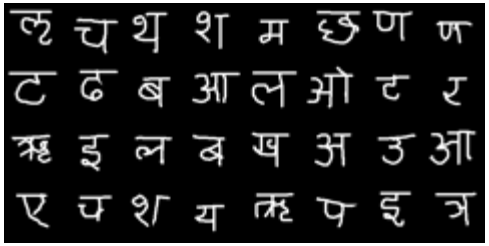
Outline

1. Course Review
2. Markov Chain Monte Carlo
3. Variational Inference
4. Example: Deep Boltzmann Machines

Deep Generative Model



Model P(image)

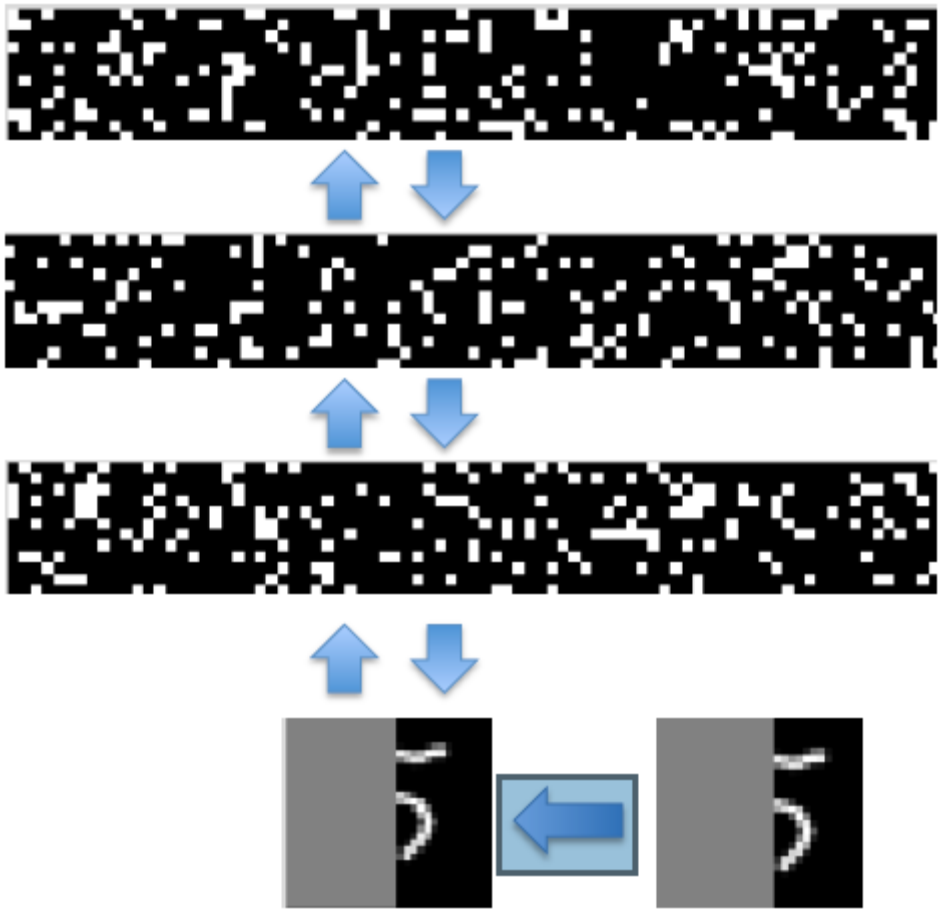


25,000 characters from 50 alphabets around the world.

- 3,000 hidden variables
- 784 observed variables (28 by 28 images)
- About 2 million parameters

Bernoulli Markov Random Field

Deep Generative Model

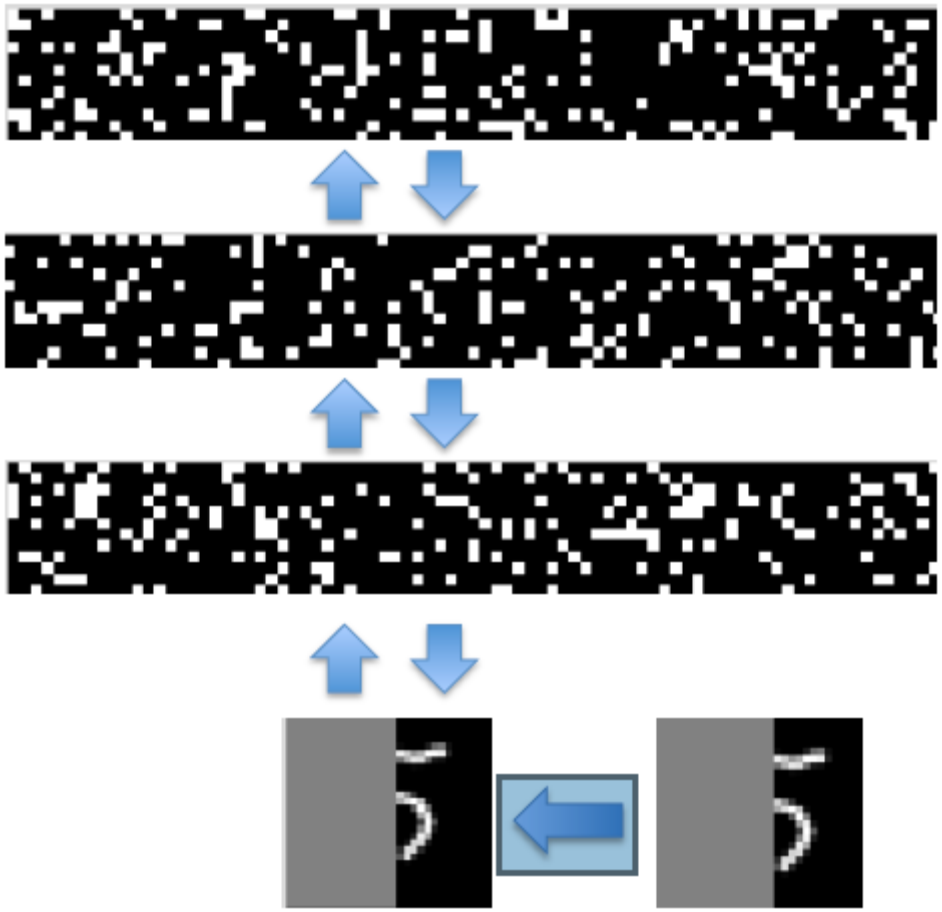


Conditional
Simulation

- P(image|partial image)

Bernoulli Markov Random Field

Deep Generative Model



Conditional
Simulation

Why so difficult?

28

28

$2^{28\times 28}$ possible images!

- P(image|partial image)

Bernoulli Markov Random Field

Fully Observed Models

- Explicitly model conditional probabilities:

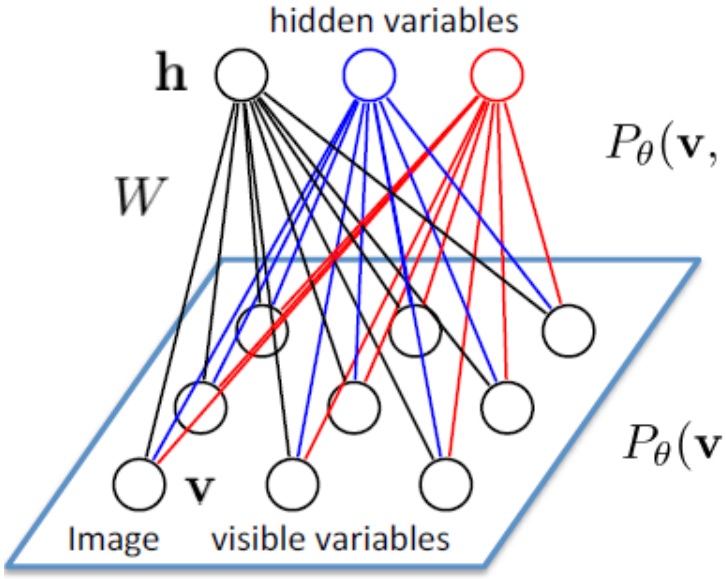
$$p_{\text{model}}(\boldsymbol{x}) = p_{\text{model}}(x_1) \prod_{i=2}^{n} p_{\text{model}}(x_i \mid x_1, \ldots, x_{i-1})$$

Each conditional can be a complicated neural network

- A number of successful models, including
  - NADE, RNADE (Larochelle, et.al. 20011)
  - Pixel CNN (van den Ord et. al. 2016)
  - Pixel RNN (van den Ord et. al. 2016)

Pixel CNN

Restricted Boltzmann Machines



Pair-wise Unary

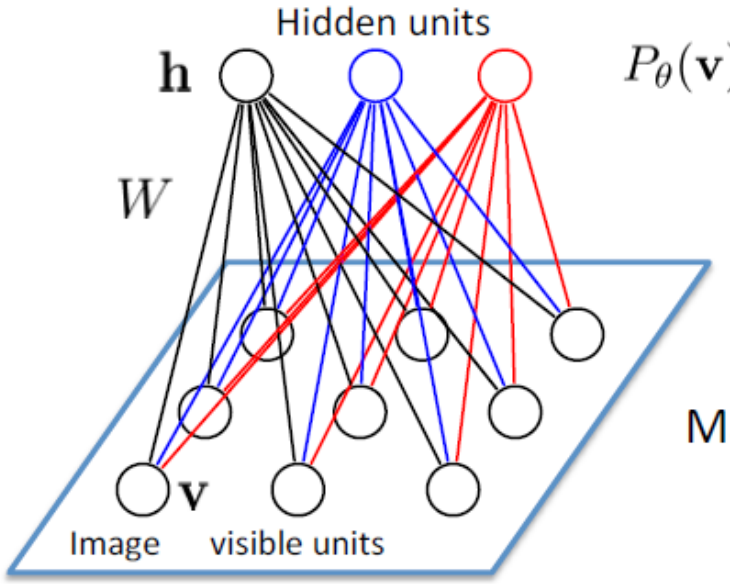
$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{i=1}^{D}\sum_{j=1}^{F} W_{ij} v_i h_j + \sum_{i=1}^{D} v_i b_i + \sum_{j=1}^{F} h_j a_j\right)$$

$$\theta = \{W, a, b\}$$

$$P_\theta(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} P_\theta(v_i|\mathbf{h}) = \prod_{i=1}^{D} \frac{1}{1+\exp(-\sum_{j=1}^{F} W_{ij} v_i h_j - b_i)}$$

- RBM is a Markov Random Field with:
- Stochastic binary hidden variables $\mathbf{v} \in \{0,1\}^D$.
- Stochastic binary visible variables $\mathbf{h} \in \{0,1\}^F$.
- Bipartite connections

Model Learning



$$P_\theta(\mathbf{v}) = \frac{P^*(\mathbf{v})}{\mathcal{Z}(\theta)} = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}} \exp\left[\mathbf{v}^\top W \mathbf{h} + \mathbf{a}^\top \mathbf{h} + \mathbf{b}^\top \mathbf{v}\right]$$

Given a set of *i.i.d.* training examples $\mathcal{D} = \{\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, ..., \mathbf{v}^{(N)}\}$, we want to learn model parameters $\theta = \{W, a, b\}$.

Maximize log-likelihood objective:

$$L(\theta) = \frac{1}{N} \sum_{n=1}^{N} \log P_\theta(\mathbf{v}^{(n)})$$

Derivative of the log-likelihood:

$$\frac{\partial L(\theta)}{\partial W_{ij}} = \frac{1}{N} \sum_{n=1}^{N} \frac{\partial}{\partial W_{ij}} \log\left(\sum_{\mathbf{h}} \exp\left[\mathbf{v}^{(n)\top} W \mathbf{h} + \mathbf{a}^\top \mathbf{h} + \mathbf{b}^\top \mathbf{v}^{(n)}\right]\right) - \frac{\partial}{\partial W_{ij}} \log \mathcal{Z}(\theta)$$

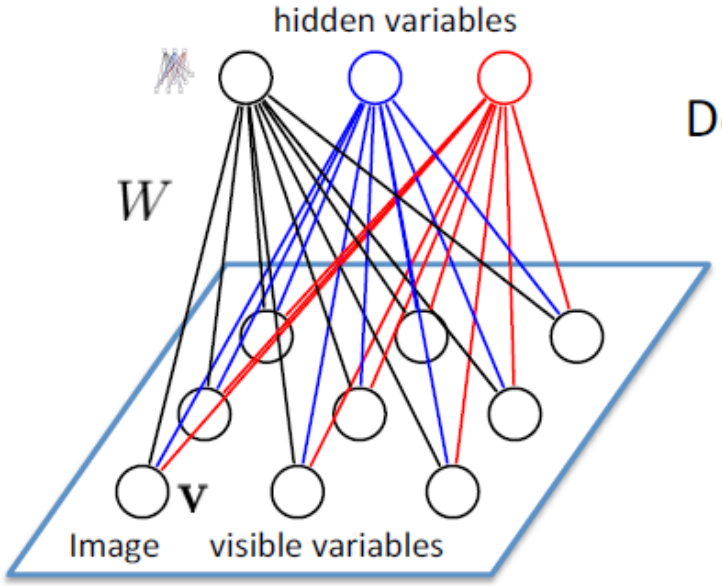
$$= \mathrm{E}_{P_{data}}[v_i h_j] - \underbrace{\mathrm{E}_{P_\theta}[v_i h_j]}$$

Difficult to compute: exponentially many configurations

$$P_{data}(\mathbf{v}, \mathbf{h}; \theta) = P(\mathbf{h}|\mathbf{v}; \theta) P_{data}(\mathbf{v})$$

$$P_{data}(\mathbf{v}) = \frac{1}{N} \sum_{n} \delta(\mathbf{v} - \mathbf{v}^{(n)})$$

Model Learning



Derivative of the log-likelihood:

$$\frac{\partial L(\theta)}{\partial W_{ij}} = \mathbb{E}_{P_{data}}[v_i h_j] - \mathbb{E}_{P_\theta}[v_i h_j]$$

$$\sum_{\mathbf{v},\mathbf{h}} v_i h_j P_\theta(\mathbf{v},\mathbf{h})$$

Easy to compute exactly

Difficult to compute: exponentially many configurations.

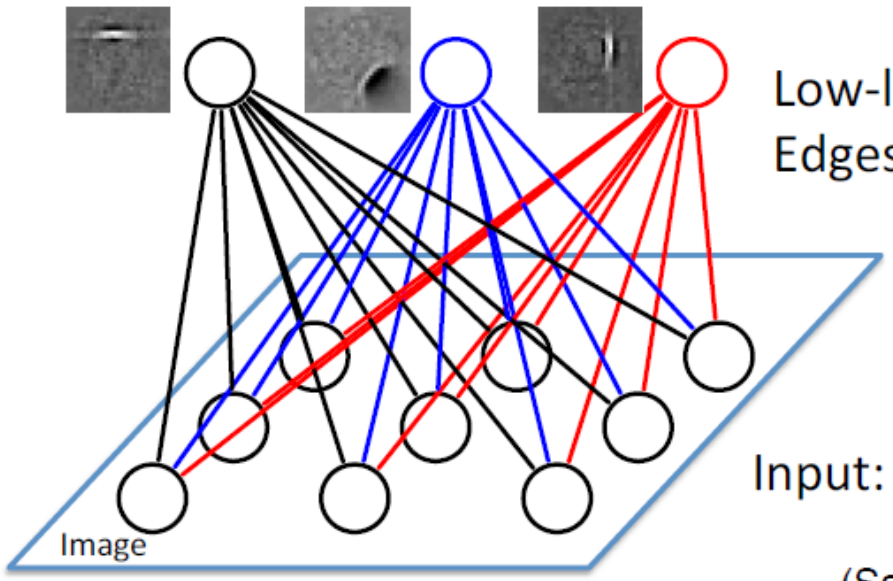
Use MCMC

$$P_{data}(\mathbf{v},\mathbf{h};\theta) = P(\mathbf{h}|\mathbf{v};\theta) P_{data}(\mathbf{v})$$

$$P_{data}(\mathbf{v}) = \frac{1}{N}\sum_n \delta(\mathbf{v} - \mathbf{v}^{(n)})$$

Approximate maximum likelihood learning

Deep Boltzmann Machines



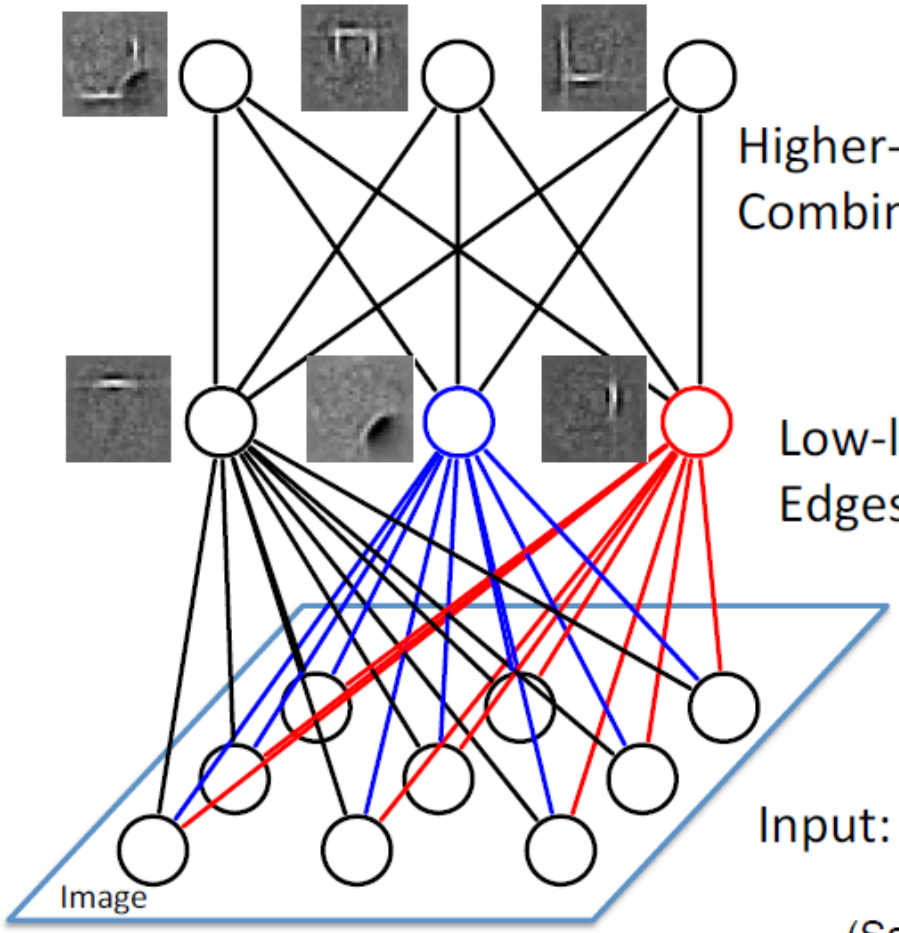
Low-level features:
Edges

Built from **unlabeled** inputs.

Input: Pixels

(Salakhutdinov 2008, Salakhutdinov & Hinton 2012)

Deep Boltzmann Machines



Learn simpler representations,
then compose more complex ones

Higher-level features:
Combination of edges

Low-level features:
Edges

Built from **unlabeled** inputs.

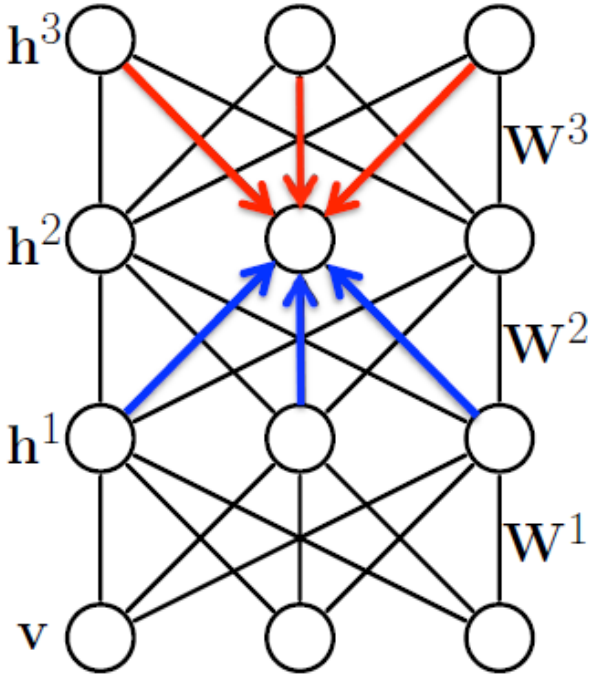
Input: Pixels

(Salakhutdinov 2008, Salakhutdinov & Hinton 2012)

Mathematical Formulation

$$P_\theta(\mathbf{v}) = \frac{P^*(\mathbf{v})}{\mathcal{Z}(\theta)} = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}^1,\mathbf{h}^2,\mathbf{h}^3} \exp\left[\mathbf{v}^\top W^1 \mathbf{h}^1 + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3\right]$$

Deep Boltzmann Machine



Input

$\theta = \{W^1, W^2, W^3\}$ model parameters

- Dependencies between hidden variables.
- All connections are undirected.
- Bottom-up and Top-down:

$$P(h_k^2 = 1 | \mathbf{h}^1, \mathbf{h}^3) = \sigma\left(\sum_j W_{jk}^2 h_j^1 + \sum_m W_{km}^3 h_m^3\right)$$

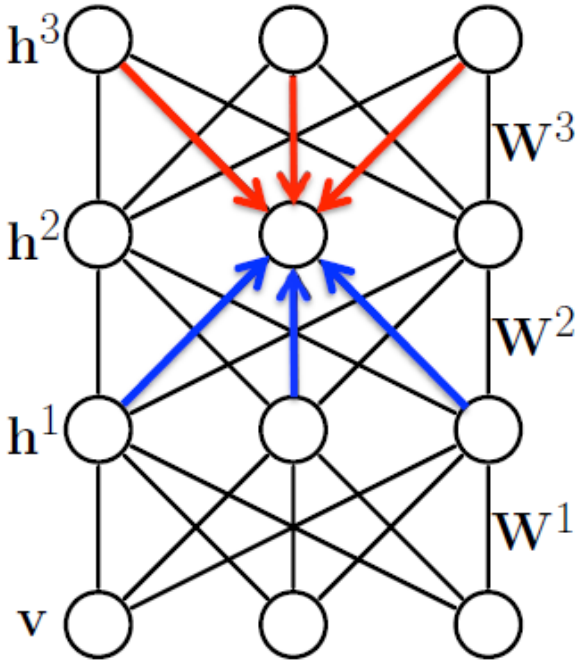
Bottom-up Top-Down

Unlike many existing feed-forward models: ConvNet (LeCun), HMAX (Poggio et.al.), Deep Belief Nets (Hinton et.al.)

Mathematical Formulation

$$P_\theta(\mathbf{v}) = \frac{P^*(\mathbf{v})}{\mathcal{Z}(\theta)} = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}^1,\mathbf{h}^2,\mathbf{h}^3} \exp\left[\mathbf{v}^\top W^1 \mathbf{h}^1 + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3\right]$$

Deep Boltzmann Machine



Input

- Conditional Distributions:

$$P(h_j^1 = 1|\mathbf{v}, \mathbf{h}^2) = \sigma\left(\sum_i W_{ij}^1 v_i + \sum_k W_{jk}^2 h_k^2\right)$$

$$P(h_k^2 = 1|\mathbf{h}^1, \mathbf{h}^3) = \sigma\left(\sum_j W_{jk}^2 h_j^1 + \sum_m W_{km}^3 h_m^3\right)$$

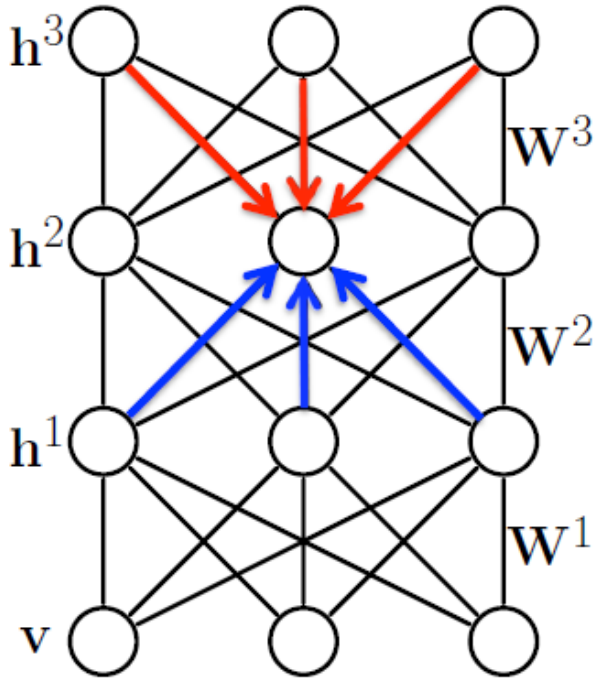
$$P(h_m^3 = 1|\mathbf{h}^2) = \sigma\left(\sum_k W_{km}^3 h_k^2\right)$$

- Note that exact computation of $P(\mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3|\mathbf{v})$ is intractable.

Mathematical Formulation

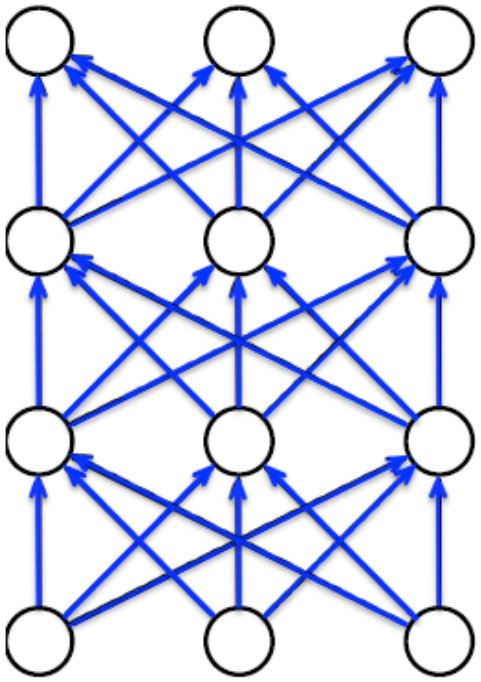
$$P_\theta(\mathbf{v}) = \frac{P^*(\mathbf{v})}{\mathcal{Z}(\theta)} = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}^1,\mathbf{h}^2,\mathbf{h}^3} \exp\left[\mathbf{v}^\top W^1 \mathbf{h}^1 + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3\right]$$

Deep Boltzmann Machine

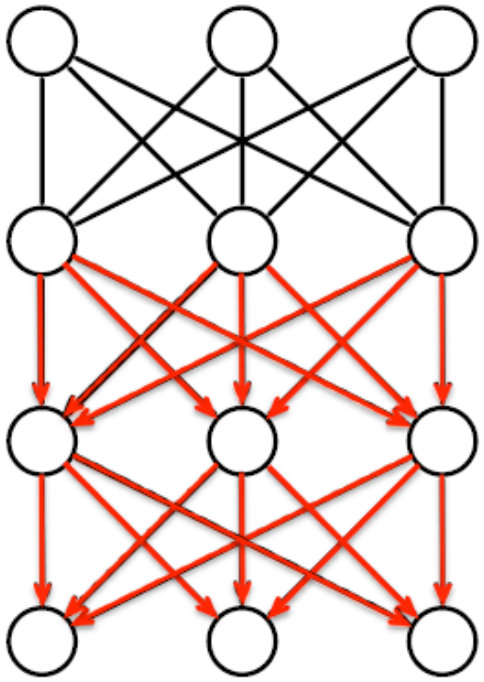


Input

Neural Network Output



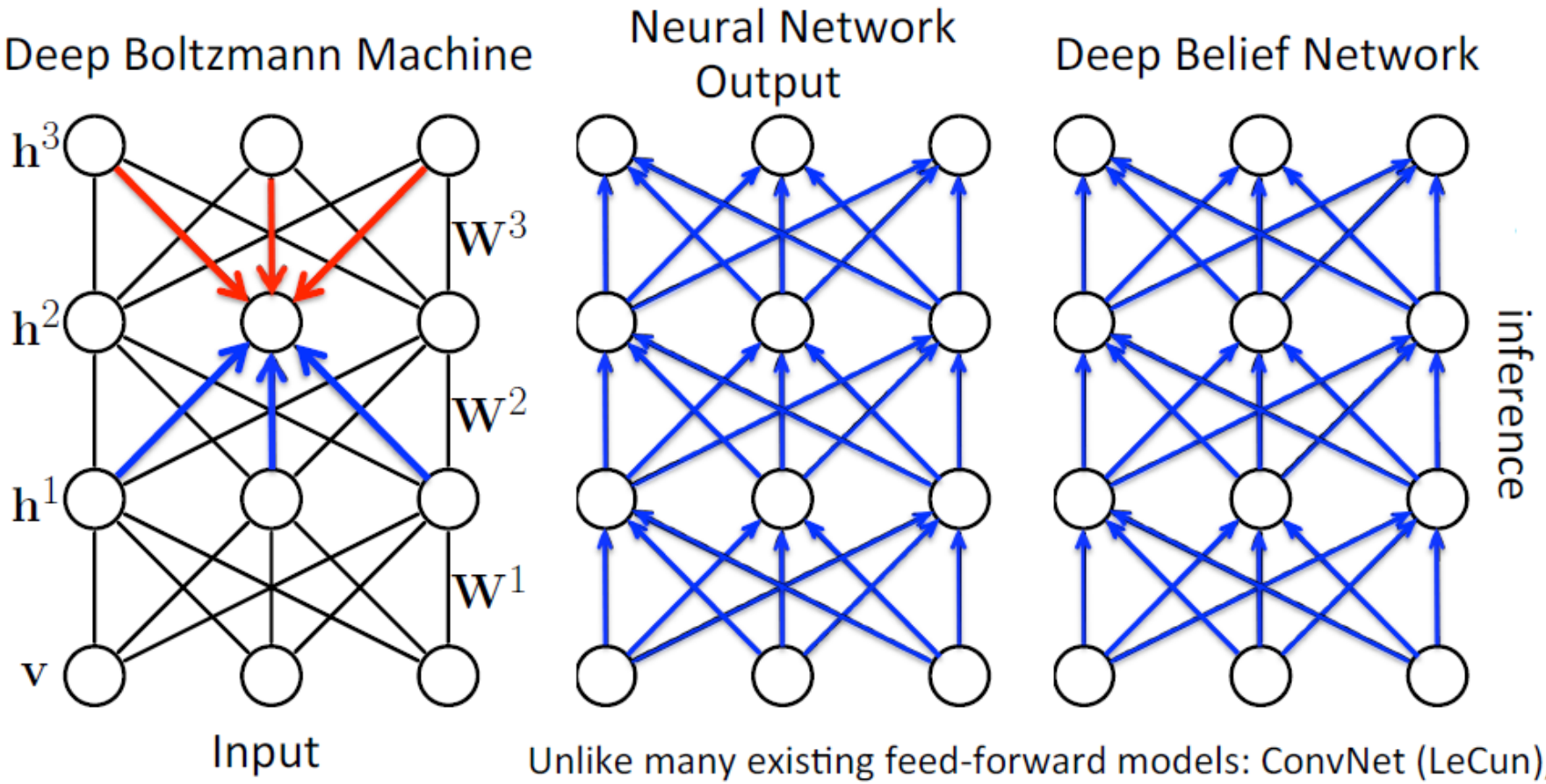
Deep Belief Network



Unlike many existing feed-forward models: ConvNet (LeCun), HMAX (Poggio), Deep Belief Nets (Hinton)

Mathematical Formulation

$$P_\theta(\mathbf{v}) = \frac{P^*(\mathbf{v})}{\mathcal{Z}(\theta)} = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}^1,\mathbf{h}^2,\mathbf{h}^3} \exp\left[\mathbf{v}^\top W^1 \mathbf{h}^1 + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3\right]$$

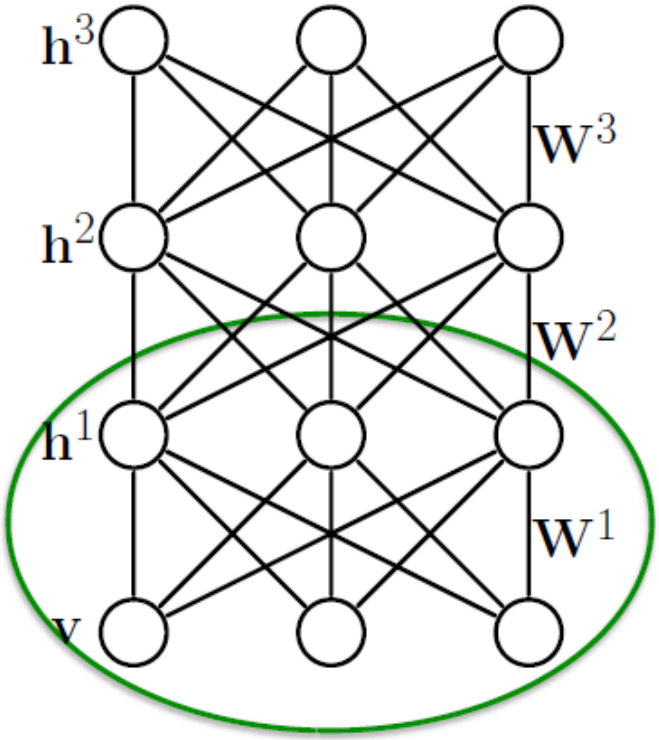


Unlike many existing feed-forward models: ConvNet (LeCun), HMAX (Poggio), Deep Belief Nets (Hinton)

Mathematical Formulation

$$P_\theta(\mathbf{v}) = \frac{P^*(\mathbf{v})}{\mathcal{Z}(\theta)} = \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}^1,\mathbf{h}^2,\mathbf{h}^3} \exp\left[\mathbf{v}^\top W^1 \mathbf{h}^1 + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3\right]$$

Deep Boltzmann Machine



$\theta = \{W^1, W^2, W^3\}$ model parameters

- Dependencies between hidden variables.

Maximum likelihood learning:

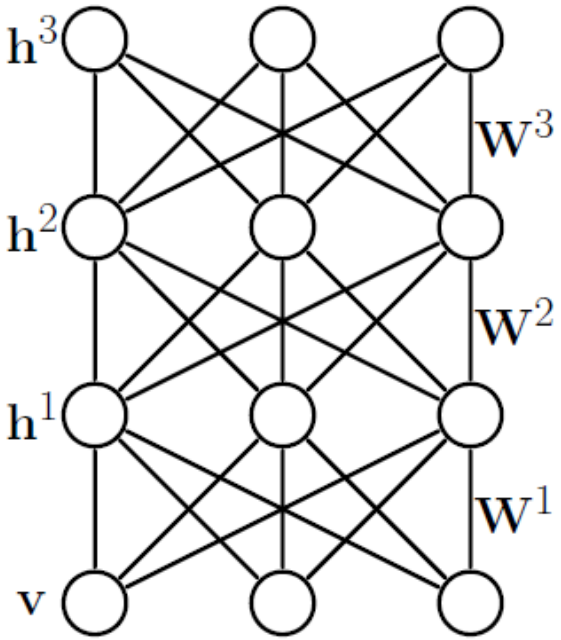
$$\frac{\partial \log P_\theta(\mathbf{v})}{\partial W^1} = \mathrm{E}_{P_{data}}[\mathbf{v}\mathbf{h}^{1\top}] - \mathrm{E}_{P_\theta}[\mathbf{v}\mathbf{h}^{1\top}]$$

Problem: Both expectations are intractable!

Learning rule for undirected graphical models: MRFs, CRFs, Factor graphs.

Approximate Learning

$$P_\theta(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left[\mathbf{v}^\top W^{(1)} \mathbf{h}^{(1)} + \mathbf{h}^{(1)\top} W^{(2)} \mathbf{h}^{(2)} + \mathbf{h}^{(2)\top} W^{(3)} \mathbf{h}^{(3)}\right]$$



(Approximate) Maximum Likelihood:

$$\frac{\partial \log P_\theta(\mathbf{v})}{\partial W^1} = \mathbb{E}_{P_{data}}[\mathbf{v}\mathbf{h}^{\mathbf{1}\top}] - \mathbb{E}_{P_\theta}[\mathbf{v}\mathbf{h}^{\mathbf{1}\top}]$$

- Both expectations are intractable!

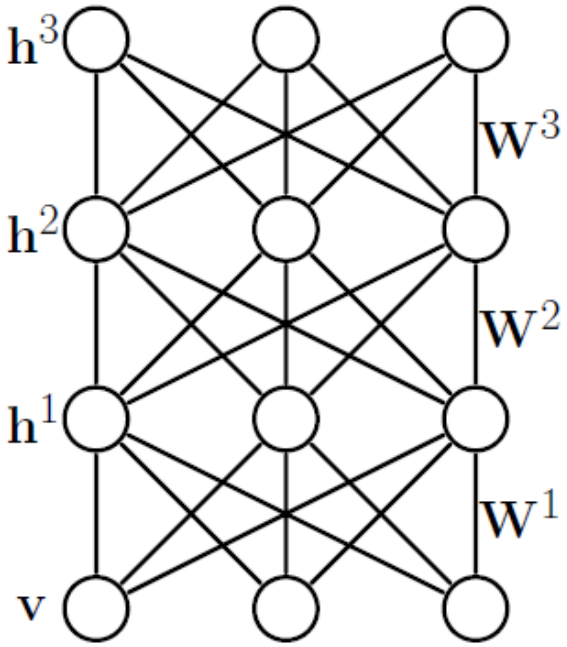
$$P_{data}(\mathbf{v}, \mathbf{h^1}) = P_\theta(\mathbf{h^1}|\mathbf{v}) P_{data}(\mathbf{v})$$

$$P_{data}(\mathbf{v}) = \frac{1}{N} \sum_{n=1}^{N} \delta(\mathbf{v} - \mathbf{v_n})$$

Not factorial any more!

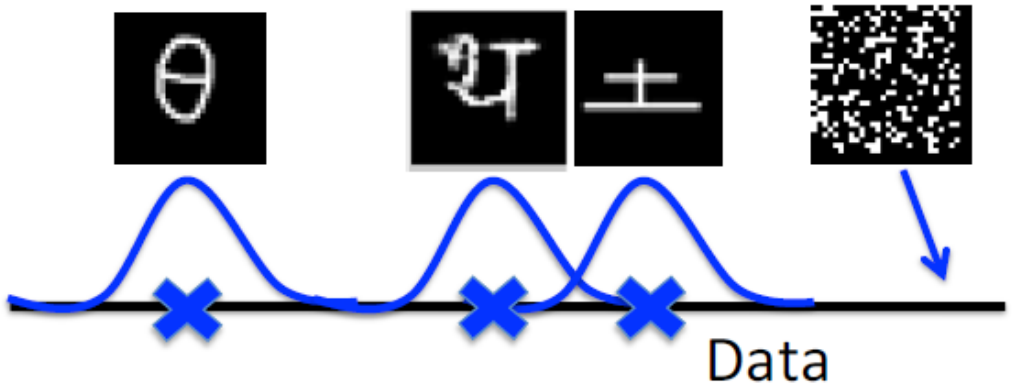
Approximate Learning

$$P_\theta(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left[\mathbf{v}^\top W^{(1)} \mathbf{h}^{(1)} + \mathbf{h}^{(1)\top} W^{(2)} \mathbf{h}^{(2)} + \mathbf{h}^{(2)\top} W^{(3)} \mathbf{h}^{(3)}\right]$$



(Approximate) Maximum Likelihood:

$$\frac{\partial \log P_\theta(\mathbf{v})}{\partial W^1} = \mathbb{E}_{P_{data}}[\mathbf{v}\mathbf{h}^{\mathbf{1}\top}] - \mathbb{E}_{P_\theta}[\mathbf{v}\mathbf{h}^{\mathbf{1}\top}]$$



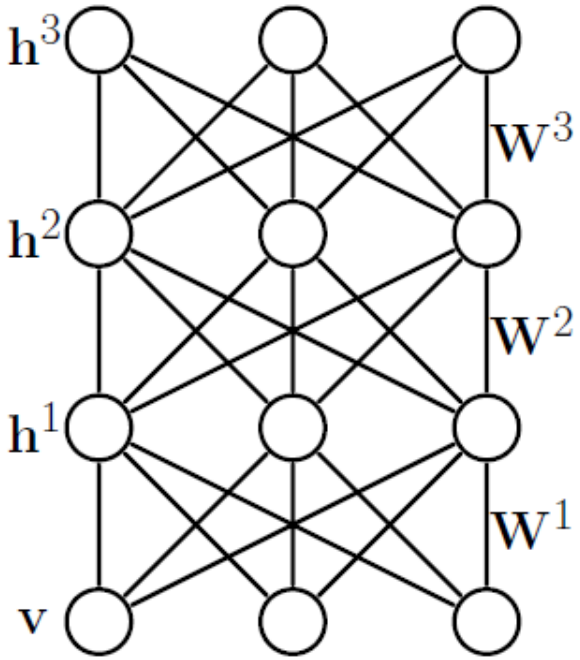
$$P_{data}(\mathbf{v}, \mathbf{h}^{\mathbf{1}}) = P_\theta(\mathbf{h}^{\mathbf{1}}|\mathbf{v}) P_{data}(\mathbf{v})$$

$$P_{data}(\mathbf{v}) = \frac{1}{N} \sum_{n=1}^{N} \delta(\mathbf{v} - \mathbf{v_n})$$

Not factorial any more!

Approximate Learning

$$P_\theta(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left[\mathbf{v}^\top W^{(1)} \mathbf{h}^{(1)} + \mathbf{h}^{(1)\top} W^{(2)} \mathbf{h}^{(2)} + \mathbf{h}^{(2)\top} W^{(3)} \mathbf{h}^{(3)}\right]$$



(Approximate) Maximum Likelihood:

$$\frac{\partial \log P_\theta(\mathbf{v})}{\partial W^1} = \mathbb{E}_{P_{data}}[\mathbf{v}\mathbf{h}^{\mathbf{1}\top}] - \mathbb{E}_{P_\theta}[\mathbf{v}\mathbf{h}^{\mathbf{1}\top}]$$

Variational Inference

Stochastic Approximation (MCMC-based)

$$P_{data}(\mathbf{v}, \mathbf{h}^{\mathbf{1}}) = P_\theta(\mathbf{h}^{\mathbf{1}}|\mathbf{v}) P_{data}(\mathbf{v})$$

$$P_{data}(\mathbf{v}) = \frac{1}{N} \sum_{n=1}^{N} \delta(\mathbf{v} - \mathbf{v_n})$$

Not factorial any more!

Previous Work

- Many approaches for learning Boltzmann machines have been proposed over the last 20 years:
  - Hinton and Sejnowski (1983),
  - Peterson and Anderson (1987)
  - Galland (1991)
  - Kappen and Rodriguez (1998)
  - Lawrence, Bishop, and Jordan (1998)
  - Tanaka (1998)
  - Welling and Hinton (2002)
  - Zhu and Liu (2002)
  - Welling and Teh (2003)
  - Yasuda and Tanaka (2009)

Real-world applications – thousands of hidden and observed variables with millions of parameters.

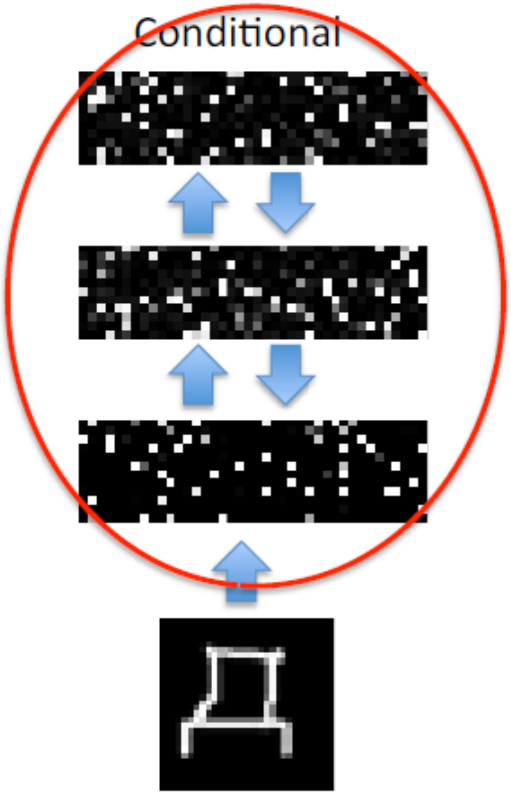
- Many of the previous approaches were not successful for learning general Boltzmann machines with **hidden variables.**

- Algorithms based on Contrastive Divergence, Score Matching, Pseudo- Likelihood, Composite Likelihood, MCMC-MLE, Piecewise Learning, cannot handle multiple layers of hidden variables.

New Learning Algorithm

Posterior Inference

Conditional



Approximate conditional

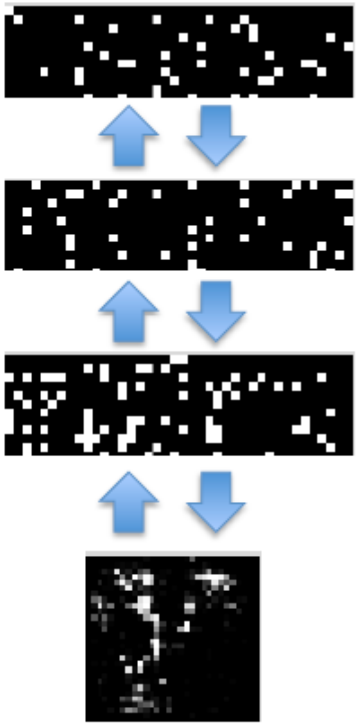
$P_{data}(\mathbf{h}|\mathbf{v})$

Simulate from the Model

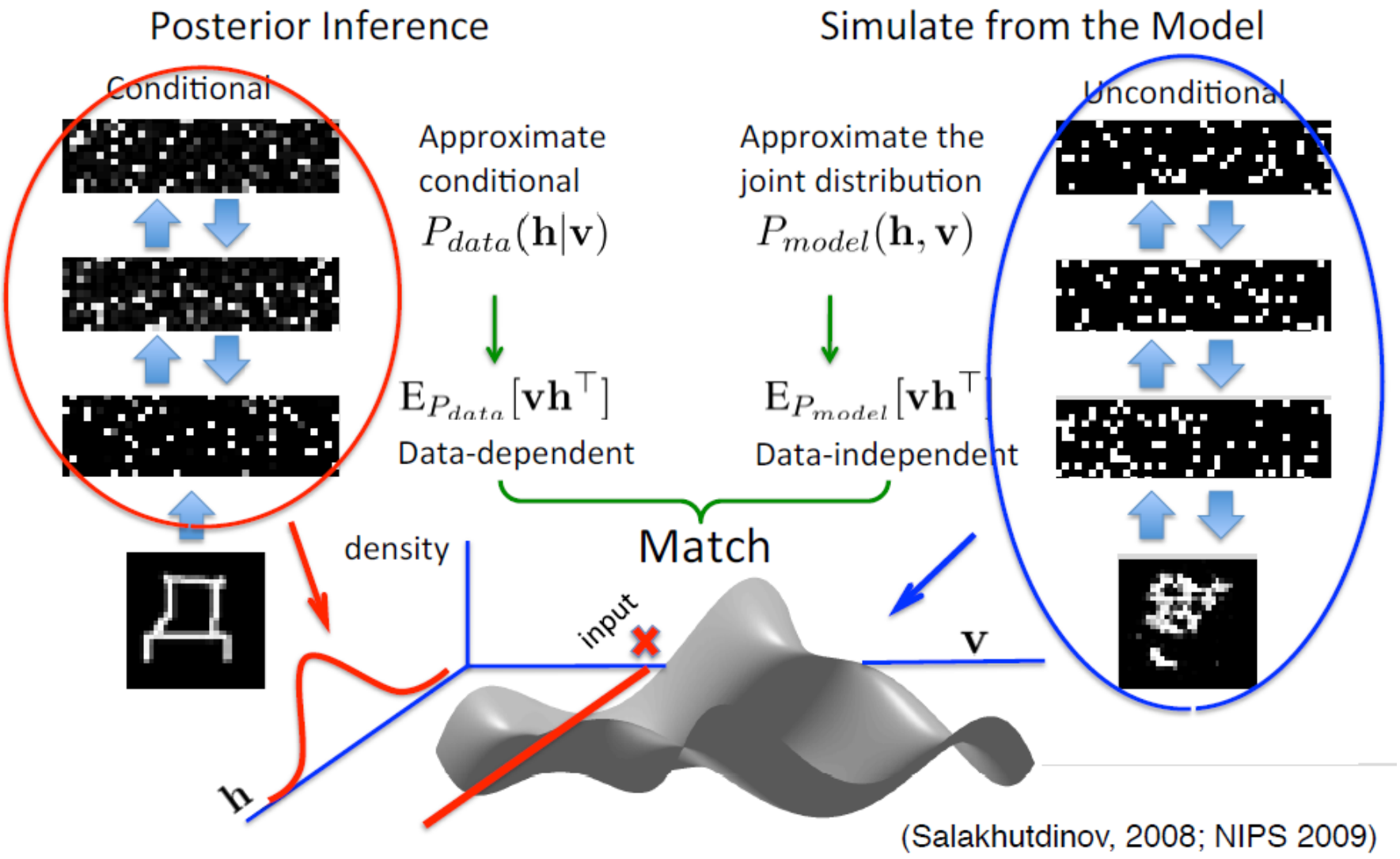
Approximate the joint distribution

$P_{model}(\mathbf{h}, \mathbf{v})$

Unconditional

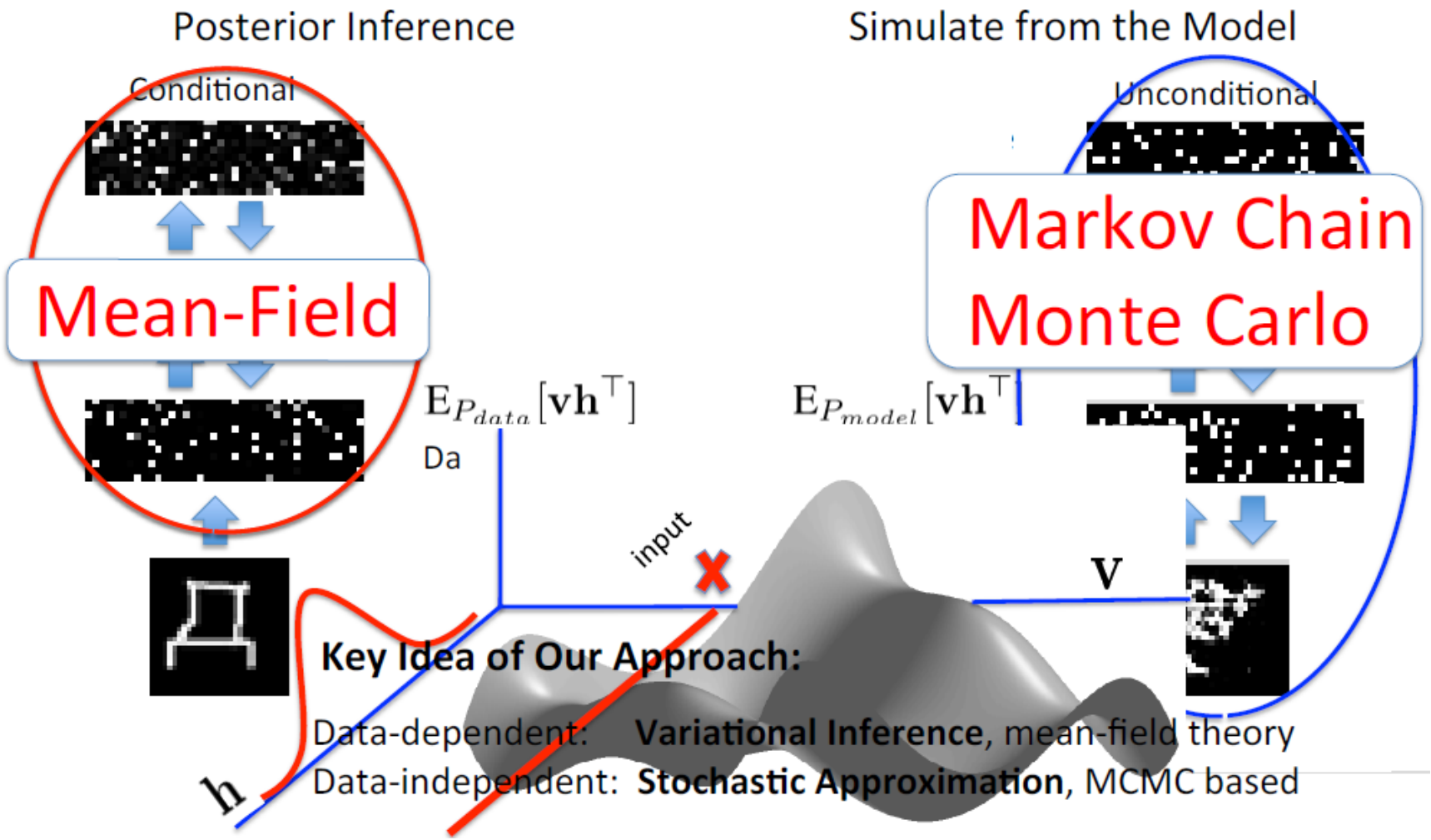


New Learning Algorithm



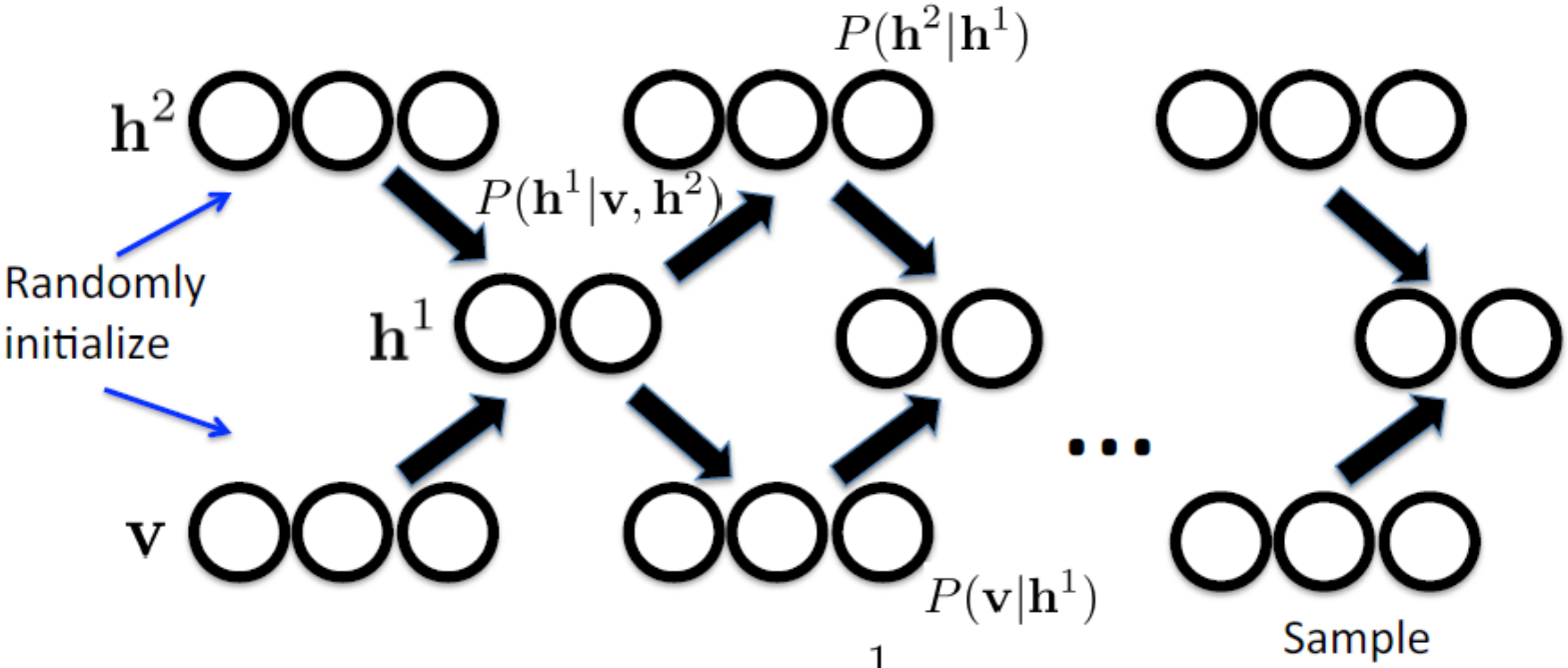
(Salakhutdinov, 2008; NIPS 2009)

New Learning Algorithm



Sampling from DBMs

Sampling from two-hidden layer DBM by running a Markov chain:

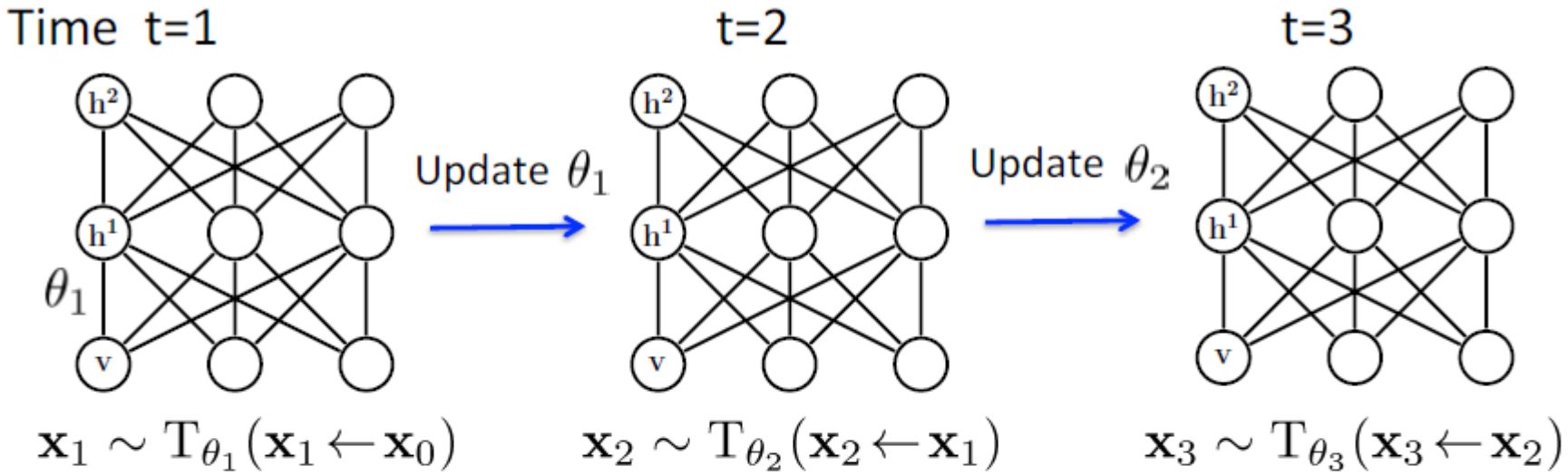


$$P(h_m^1 = 1|\mathbf{v},\mathbf{h}^2) = \frac{1}{1+\exp(-\sum_i W_{im}^1 v_i - \sum_j W_{mj}^2 h_j^2)}$$

$$P(h_j^2 = 1|\mathbf{h}^1) = \frac{1}{1+\exp(-\sum_m W_{mj}^2 h_m^1)}$$

$$P(v_i = 1|\mathbf{h}^1) = \frac{1}{1+\exp(-\sum_m W_{im}^1 h_m^1)}$$

Stochastic Approximation



Update θ_t and $\mathbf{x}_t$ sequentially, where $\mathbf{x} = \{\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2\}$

- Generate $\mathbf{x}_t \sim \mathrm{T}_{\theta_t}(\mathbf{x}_t \leftarrow \mathbf{x}_{t-1})$ by simulating from a Markov chain that leaves P_{θ_t} invariant (e.g. Gibbs or M-H sampler)
- Update θ_t by replacing intractable $\mathrm{E}_{P_{\theta_t}}[\mathbf{v}\mathbf{h}^\top]$ with a point estimate $[\mathbf{v}_t\mathbf{h}_t^\top]$

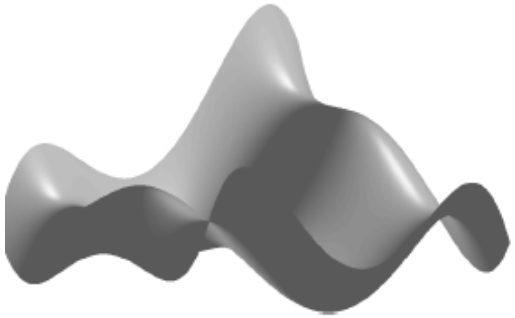
In practice we simulate several Markov chains in parallel.

Learning Algorithm

Update rule decomposes:

$$\theta_{t+1} = \theta_t + \alpha_t \underbrace{\left(\mathbb{E}_{P_{data}}[\mathbf{v}\mathbf{h}^\top] - \mathbb{E}_{P_{\theta_t}}[\mathbf{v}\mathbf{h}^\top] \right)}_{\text{True gradient}} + \alpha_t \underbrace{\left(\mathbb{E}_{P_{\theta_t}}[\mathbf{v}\mathbf{h}^\top] - \frac{1}{M} \sum_{m=1}^{M} \mathbf{v}_t^{(m)} \mathbf{h}_t^{(m)\top} \right)}_{\text{Perturbation term } \epsilon_t}$$

Almost sure convergence guarantees as learning rate $\alpha_t \to 0$



Problem: High-dimensional data: the probability landscape is highly multimodal.

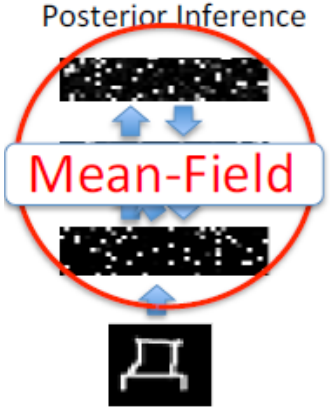
Key insight: The transition operator can b
any valid transition operator – Tempered
Transitions, Parallel/Simulated Tempering

Connections to the theory of stochastic approximation and adaptive MCMC.

Variational Inference

Approximate intractable distribution $P_\theta(\mathbf{h}|\mathbf{v})$ with simpler, tractable distribution $Q_\mu(\mathbf{h}|\mathbf{v})$:

$$\log P_\theta(\mathbf{v}) = \log \sum_{\mathbf{h}} P_\theta(\mathbf{h}, \mathbf{v}) = \log \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \frac{P_\theta(\mathbf{h}, \mathbf{v})}{Q_\mu(\mathbf{h}|\mathbf{v})}$$



$$\geq \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \log \frac{P_\theta(\mathbf{h}, \mathbf{v})}{Q_\mu(\mathbf{h}|\mathbf{v})}$$

$$= \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \underbrace{\log P^*_\theta(\mathbf{h}, \mathbf{v})}_{\mathbf{v}^\top W^1 \mathbf{h}^1 + \mathbf{h}^{1\top} W^2 \mathbf{h}^2 + \mathbf{h}^{2\top} W^3 \mathbf{h}^3} - \log \mathcal{Z}(\theta) + \sum_{\mathbf{h}} Q_\mu(\mathbf{h}|\mathbf{v}) \log \frac{1}{Q_\mu(\mathbf{h}|\mathbf{v})}$$

Variational Lower Bound

$$= \log P_\theta(\mathbf{v}) - \mathrm{KL}\big(Q_\mu(\mathbf{h}|\mathbf{v}) || P_\theta(\mathbf{h}|\mathbf{v})\big)$$

$$\mathrm{KL}(Q||P) = \int Q(x) \log \frac{Q(x)}{P(x)} dx$$

Minimize KL between approximating and true distributions with respect to variational parameters μ.

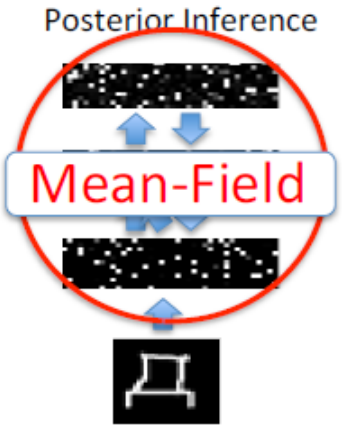
(Salakhutdinov, 2008; Salakhutdinov & Larochelle, AI & Statistics 2010)

Variational Inference

Approximate intractable distribution $P_\theta(\mathbf{h}|\mathbf{v})$ with simpler, tractable distribution $Q_\mu(\mathbf{h}|\mathbf{v})$:

$$\mathrm{KL}(Q||P) = \int Q(x) \log \frac{Q(x)}{P(x)} dx$$

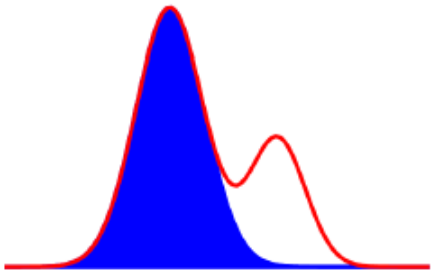
$$\log P_\theta(\mathbf{v}) \geq \underbrace{\log P_\theta(\mathbf{v}) - \mathbf{KL}\big(Q_\mu(\mathbf{h}|\mathbf{v})||P_\theta(\mathbf{h}|\mathbf{v})\big)}_{\text{Variational Lower Bound}}$$



Mean-Field: Choose a fully factorized distribution:

$$Q_\mu(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{F} q(h_j|\mathbf{v}) \text{ with } q(h_j = 1|\mathbf{v}) = \mu_j$$

Variational Inference: Maximize the lower bound w.r.t. Variational parameters μ.



Nonlinear fixed-point equations:

$$\mu_j^{(1)} = \sigma\Big(\sum_i W_{ij}^1 v_i + \sum_k W_{jk}^2 \mu_k^{(2)}\Big)$$
$$\mu_k^{(2)} = \sigma\Big(\sum_j W_{jk}^2 \mu_j^{(1)} + \sum_m W_{km}^3 \mu_m^{(3)}\Big)$$
$$\mu_m^{(3)} = \sigma\Big(\sum_k W_{km}^3 \mu_k^{(2)}\Big)$$

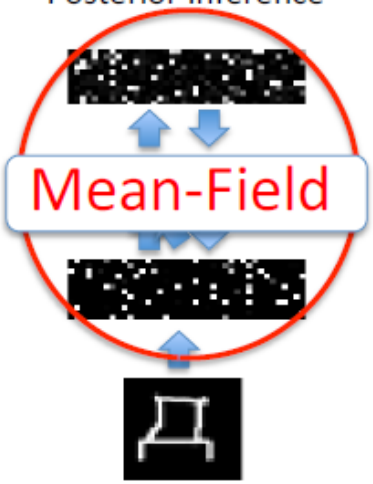
Variational Inference

Approximate intractable distribution $P_\theta(\mathbf{h}|\mathbf{v})$ with simpler, tractable distribution $Q_\mu(\mathbf{h}|\mathbf{v})$:

$$\mathrm{KL}(Q||P) = \int Q(x) \log \frac{Q(x)}{P(x)} dx$$

$$\log P_\theta(\mathbf{v}) \geq \underbrace{\log P_\theta(\mathbf{v}) - \mathrm{KL}\big(Q_\mu(\mathbf{h}|\mathbf{v})||P_\theta(\mathbf{h}|\mathbf{v})\big)}_{\text{Variational Lower Bound}}$$

Posterior Inference



1. Variational Inference: Maximize the lower bound w.r.t. variational parameters

2. MCMC: Apply stochastic approximation to update model parameters

Unconditional Simulation

Almost sure convergence guarantees to an asymptotically stable point.

Variational Inference

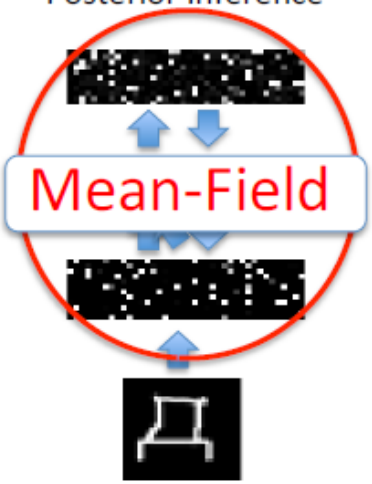
Approximate intractable distribution $P_\theta(\mathbf{h}|\mathbf{v})$ with simpler, tractable distribution $Q_\mu(\mathbf{h}|\mathbf{v})$:

$$\mathrm{KL}(Q||P) = \int Q(x) \log \frac{Q(x)}{P(x)} dx$$

$$\log P_\theta(\mathbf{v}) \geq \log P_\theta(\mathbf{v}) - \mathrm{KL}\big(Q_\mu(\mathbf{h}|\mathbf{v})||P_\theta(\mathbf{h}|\mathbf{v})\big)$$

Variational Lower Bound

Posterior Inference



Unconditional Simulation

1. V... bou... wer

Fast Inference

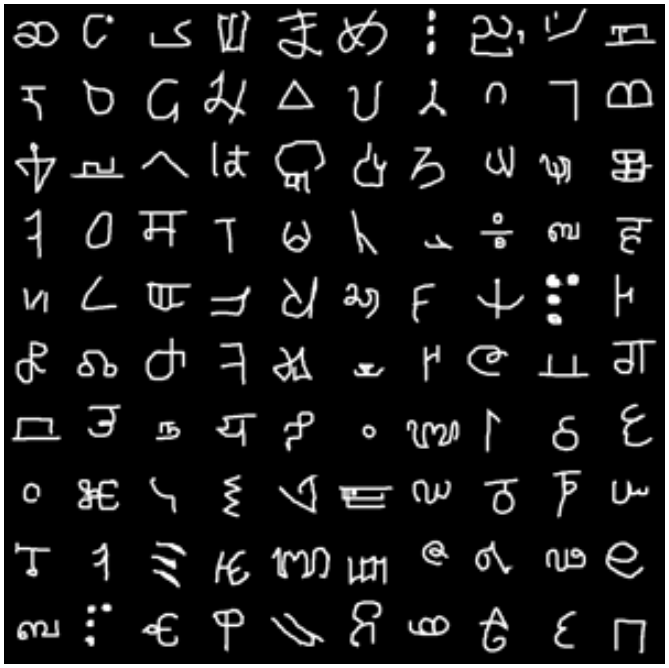
2. M... to u...

Learning can scale to millions of examples

Almost sure convergence guarantees to an asymptotically stable point.

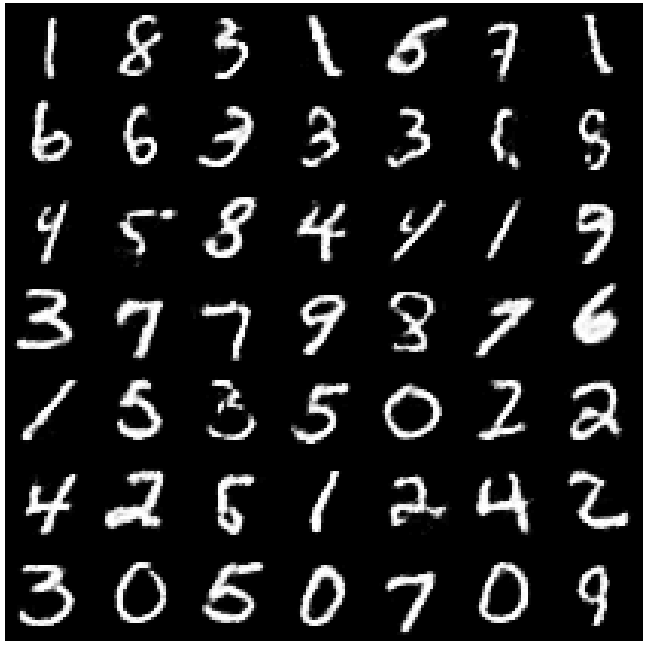
Good Generative Model?

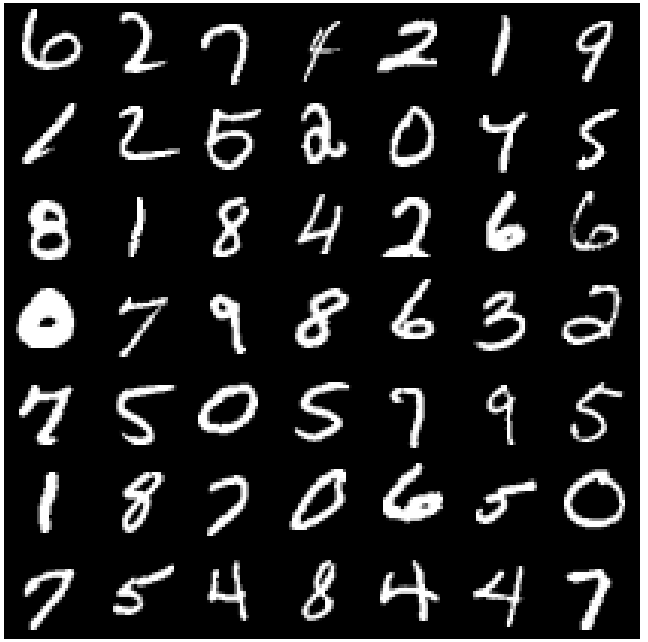
- Handwritten Characters



Good Generative Model?

- MNIST Handwritten Digit Dataset





Handwriting Recognition

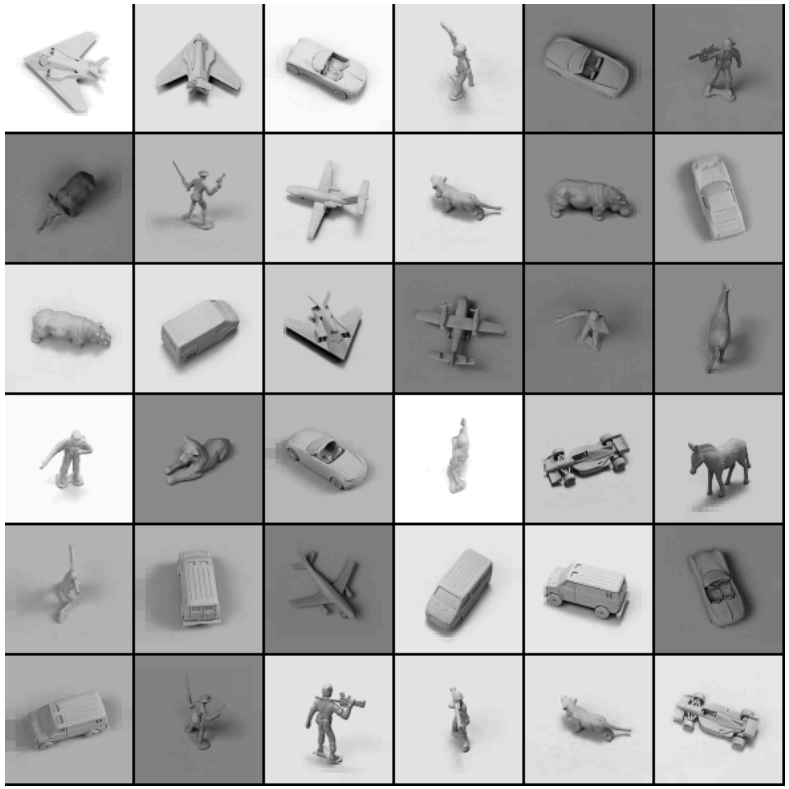
MNIST Dataset
60,000 examples of 10 digits

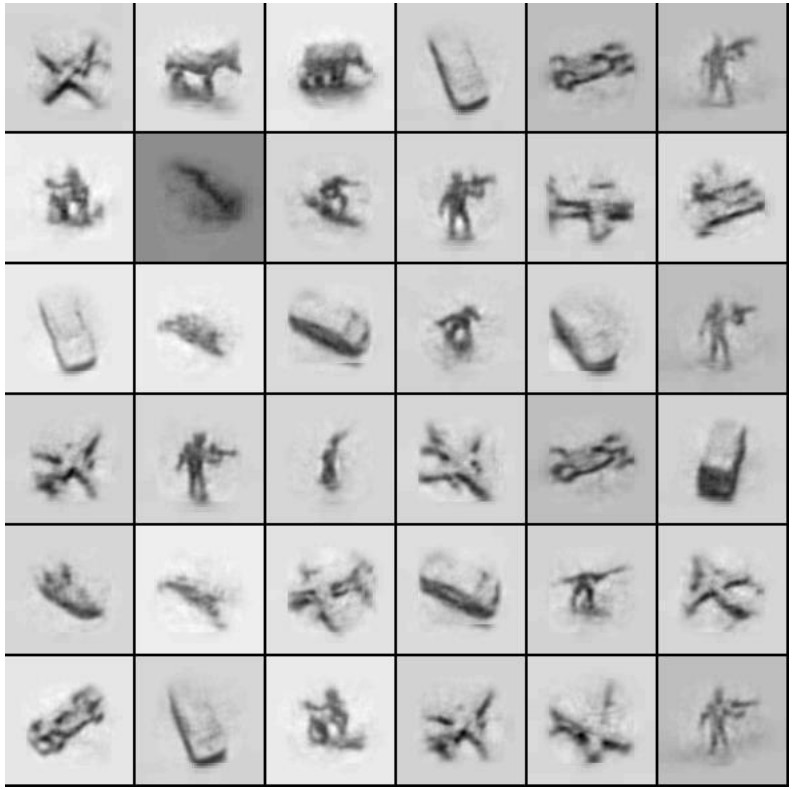
Learning Algorithm	Error
Logistic regression	12.0%
K-NN	3.09%
Neural Net (Platt 2005)	1.53%
SVM (Decoste et.al. 2002)	1.40%
Deep Autoencoder (Bengio et. al. 2007)	1.40%
Deep Belief Net (Hinton et. al. 2006)	1.20%
DBM	**0.95%**

Optical Character Recognition
42,152 examples of 26 English letters

Learning Algorithm	Error
Logistic regression	22.14%
K-NN	18.92%
Neural Net	14.62%
SVM (Larochelle et.al. 2009)	9.70%
Deep Autoencoder (Bengio et. al. 2007)	10.05%
Deep Belief Net (Larochelle et. al. 2009)	9.68%
DBM	**8.40%**

Generative Model of 3-D Objects



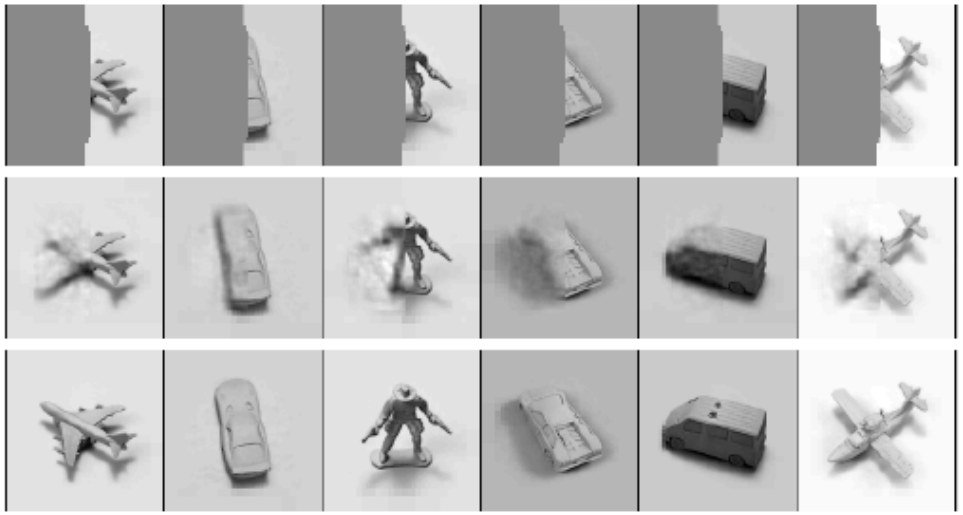


- 24,000 examples, 5 object categories, 5 different objects within each category, 6 lightning conditions, 9 elevations, 18 azimuths.

3-D Object Recognition

Learning Algorithm	Error
Logistic regression	22.5%
K-NN (LeCun 2004)	18.92%
SVM (Bengio & LeCun 2007)	11.6%
Deep Belief Net (Nair & Hinton 2009)	9.0%
DBM	**7.2%**

Pattern Completion

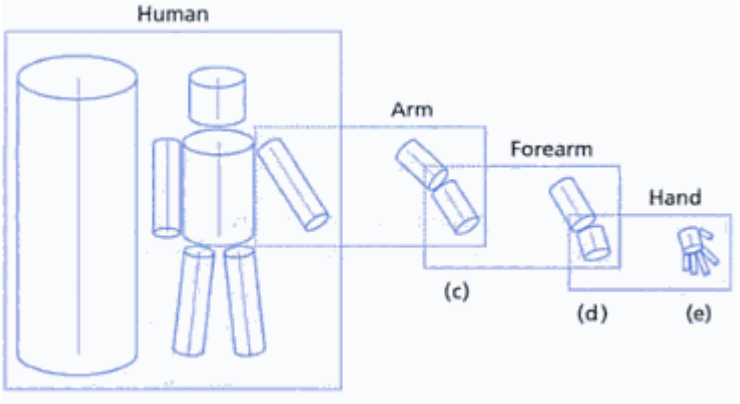


Learning Hierarchical Representations

- Deep Boltzmann Machines:

  Learning Hierarchical Structure in Features: edges, combination of edges.

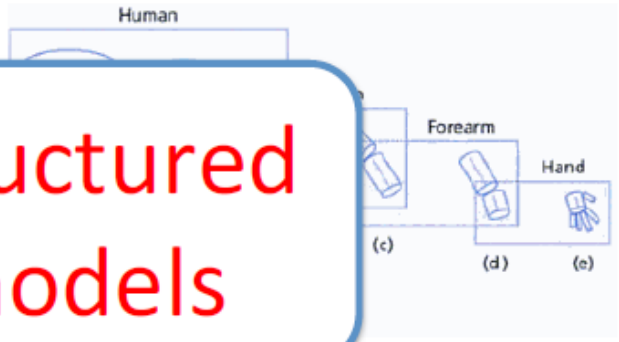
  - Performs well in many application domains
  - Fast Inference: fraction of a second
  - Learning scales to millions of examples



Learning Hierarchical Representations

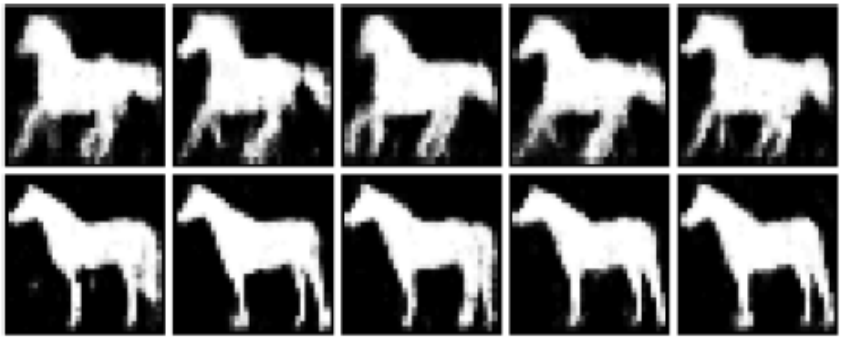
Deep Boltzmann Machines:

Learning Hi
in Features
of edges.



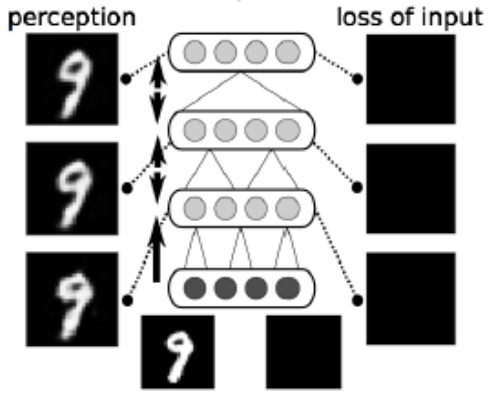
Need more structured and robust models

The Shape Boltzmann Machine: a Strong Model of Object Shape
(Eslami, Heess, Winn, CVPR 2012).



Demo DBM

Hallucinations in Charles Bonnet Syndrome Induced by Homeostasis: a Deep Boltzmann Machine Model
(Reichert, Series, Storkey, NIPS 2012)



Acknowledgement

Some of the materials in these slides are drawn inspiration from:

- Shubhendu Trivedi and Risi Kondor, University of Chicago, Deep Learning Course

- Hung-yi Lee, National Taiwan University, Machine Learning and having it Deep and Structured course

- Xiaogang Wang, The Chinese University of Hong Kong, Deep Learning Course

- Fei-Fei Li, Standord University, CS231n Convolutional Neural Networks for Visual Recognition course

Next time

- Deep Generative Model (2)

Questions?

Thank You !

WeChat Group for Deep Learning