Deep Learning

Russ Salakhutdinov

Deep Learning Summer School

Department of Computer Science
University of Toronto → CMU
Canadian Institute for Advanced Research
Current Student and Postdocs

**PhD Students**
- Lei Jimmy Ba
- Ryan Kiros
- Chris Maddison
- Nitish Srivastava
- Charlie Tang

**Postdocs**
- Yura Burda
- Roger Grosse
- Shikhar Sharma

**Master Students**
- Yukun Zhu

**Undergrads**
- Emilio Parisotto
- Elman Mansimov
Mining for Structure

Massive increase in both computational power and the amount of data available from web, video cameras, laboratory measurements.

Images & Video
- flickr
- Google
- YouTube

Text & Language
- Wikipedia
- Reuters
- Associated Press

Speech & Audio
- fMRI
- Tumor region

Relational Data/Social Network
- Amazon
- Netflix
- eBay
- Facebook
- Twitter

Mostly Unlabeled
- Develop statistical models that can discover underlying structure, cause, or statistical correlation from data in **unsupervised** or **semi-supervised** way.
- Multiple application domains.
Mining for Structure

Massive increase in both computational power and the amount of data available from web, video cameras, laboratory measurements.

Images & Video

Text & Language

Speech & Audio

Gene Expression

Deep Learning Models that support inferences and discover structure at multiple levels.

Mostly Unlabeled

- Develop statistical models that can discover underlying structure, cause, or statistical correlation from data in **unsupervised** or **semi-supervised** way.
- Multiple application domains.
Deep Generative Model

Model P(document)

Bag of words

Reuters dataset: 804,414 newswire stories: unsupervised

Interbank Markets
European Community Monetary/Economic
Energy Markets
Disasters and Accidents
Legal/Judicial
Government Borrowings
Leading Economic Indicators
Accounts/Earnings

(Hinton & Salakhutdinov, Science 2006)
Multimodal Data

mosque, tower, building, cathedral, dome, castle

kitchen, stove, oven, refrigerator, microwave

ski, skiing, skiers, skiiers, snowmobile

bowl, cup, soup, cups, coffee

beach

snow
Example: Understanding Images

TAGS:
strangers, coworkers, conventioneers, attendants, patrons

Nearest Neighbor Sentence:
people taking pictures of a crazy person

Model Samples
• a group of people in a crowded area.
• a group of people are walking and talking.
• a group of people, standing around and talking.
• a group of people that are in the outside.
Caption Generation

a car is parked in the middle of nowhere.

a wooden table and chairs arranged in a room.

there is a cat sitting on a shelf.

a ferry boat on a marina with a group of people.

a little boy with a bunch of friends on the street.
Talk Roadmap

• Learning Deep Models
  – Restricted Boltzmann Machines
  – Deep Boltzmann Machines

• Multi-Modal Learning with DBMs

• Evaluating Deep Generative Models
Learning Feature Representations

Input Space

Learning Algorithm

Segway
Non-Segway
Learning Feature Representations

Input Space

Feature Space

Segway
Non-Segway

Learning Algorithm
Traditional Approaches

Data -> Feature extraction -> Learning algorithm

Object detection

Image -> vision features -> Recognition

Audio classification

Audio -> audio features -> Speaker identification
Computer Vision Features

- SIFT
- HoG
- Textons
- GIST
- RIFT
Computer Vision Features

Deep Learning
Audio Features

Spectrogram

MFCC

Flux

ZCR

Rolloff
Audio Features

- Flux
- ZCR
- Rolloff

Deep Learning
Restricted Boltzmann Machines

RBM is a Markov Random Field with:

• Stochastic binary visible variables \( \mathbf{v} \in \{0, 1\}^D \).
• Stochastic binary hidden variables \( \mathbf{h} \in \{0, 1\}^F \).
• Bipartite connections.

Markov random fields, Boltzmann machines, log-linear models.

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

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

Pair-wise

Unary
Learning Features

Observed Data
Subset of 25,000 characters

Learned W: “edges”
Subset of 1000 features

New Image:

\[ p(h_7 = 1|v) \]
\[ = \sigma \left( 0.99 \times \right) \]
\[ + 0.97 \times \]
\[ + 0.82 \times \]

\[ \sigma(x) = \frac{1}{1 + \exp(-x)} \]
Logistic Function: Suitable for modeling binary images

Sparse representations
Model Learning

\[ P_\theta(v) = \frac{P^*(v)}{\mathcal{Z}(\theta)} = \frac{1}{\mathcal{Z}(\theta)} \sum_h \exp \left[ v^T Wh + a^T h + b^T v \right] \]

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

Maximize log-likelihood objective:

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

Derivative of the log-likelihood:

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

\[ = E_{P_{data}}[v_i h_j] - E_{P_\theta}[v_i h_j] \]

\[ P_{data}(v, h; \theta) = P(h|v; \theta)P_{data}(v) \]

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

Difficult to compute: exponentially many configurations
Model Learning

Derivative of the log-likelihood:

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

Easy to compute exactly

Difficult to compute: exponentially many configurations.
Use MCMC

Approximate maximum likelihood learning

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

$$P_{data}(v) = \frac{1}{N} \sum_{n} \delta(v - v^{(n)})$$
RBMs for Real-valued Data

\[ P_\theta(v, h) = \frac{1}{Z(\theta)} \exp \left( \sum_{i=1}^{D} \sum_{j=1}^{F} W_{ij} h_j \frac{v_i}{\sigma_i} + \sum_{i=1}^{D} \frac{(v_i - b_i)^2}{2\sigma_i^2} + \sum_{j=1}^{F} a_j h_j \right) \]

\[ P_\theta(v|h) = \prod_{i=1}^{D} P_{\theta}(v_i|h) = \prod_{i=1}^{D} \mathcal{N} \left( b_i + \sum_{j=1}^{F} W_{ij} h_j, \sigma_i^2 \right) \]

Gaussian-Bernoulli RBM:

- Stochastic real-valued visible variables \( v \in \mathbb{R}^D \).
- Stochastic binary hidden variables \( h \in \{0, 1\}^F \).
- Bipartite connections.

(Salakhutdinov & Hinton, NIPS 2007; Salakhutdinov & Murray, ICML 2008)
**RBMbs for Real-valued Data**

\[ P_{\theta}(\mathbf{v}, \mathbf{h}) = \frac{1}{Z(\theta)} \exp \left( \sum_{i=1}^{D} \sum_{j=1}^{F} W_{ij} h_j \frac{v_i}{\sigma_i} + \sum_{i=1}^{D} \frac{(v_i - b_i)^2}{2\sigma_i^2} + \sum_{j=1}^{F} a_j h_j \right) \]

\[ \theta = \{ W, a, b \} \]

\[ P_{\theta}(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} P_{\theta}(v_i|h) = \prod_{i=1}^{D} \mathcal{N} \left( b_i + \sum_{j=1}^{F} W_{ij} h_j, \sigma_i^2 \right) \]

*4 million unlabelled images*  

Learned features (out of 10,000)
RBMs for Real-valued Data

4 million *unlabelled* images

New Image

\[ p(h_7 = 1|v) = 0.9 \ast \]

\[ p(h_{29} = 1|v) = 0.8 \ast \]

...
RBMs for Word Counts

Replicated Softmax Model: undirected topic model:

- Stochastic 1-of-K visible variables.
- Stochastic binary hidden variables $\mathbf{h} \in \{0, 1\}^F$.
- Bipartite connections.

(Salakhutdinov & Hinton, NIPS 2010, Srivastava & Salakhutdinov, NIPS 2012)
RBMs for Word Counts

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

\[
P_\theta(v_i^k = 1|h) = \frac{\exp \left( b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k \right)}{\sum_{q=1}^{K} \exp \left( b_i^q + \sum_{j=1}^{F} h_j W_{ij}^q \right)}
\]

\[\theta = \{W, a, b\}\]

Learned features: "topics"

Reuters dataset: 804,414 unlabeled newswire stories

Bag-of-Words

- russian
- russia
- moscow
- yeltsin
- soviet
- clinton
- house
- president
- bill
- congress
- computer
- system
- product
- software
- develop
- trade
- country
- import
- world
- economy
- stock
- wall street
- point dow
Different Data Modalities

- Binary/Gaussian/Softmax RBMs: All have binary hidden variables but use them to model different kinds of data.

- It is easy to infer the states of the hidden variables:

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

The joint distribution is given by:

\[ P_\theta(v, h) = \frac{1}{\mathcal{Z}(\theta)} \exp \left( \sum_{ij} W_{ij}v_i h_j + \sum_i b_i v_i + \sum_j a_j h_j \right) \]

Marginalizing over hidden variables:

\[ P_\theta(v) = \sum_h P_\theta(v, h) = \frac{1}{\mathcal{Z}(\theta)} \prod_i \exp(b_i v_i) \prod_j \left( 1 + \exp(a_j + \sum_i W_{ij}v_i) \right) \]

Topics “government”, “corruption” and “oil” can combine to give very high probability to a word “Putin”.
Product of Experts

The joint distribution is given by:

\[ P_\theta(v, h) = \frac{1}{Z(\theta)} \exp \left( \sum_{ij} W_{ij} v_i h_j + \sum_i b_i v_i + \sum_j a_j h_j \right) \]

Marginalizing over hidden variables:

\[ P_\theta(v) = \sum_h P_\theta(v, h) \]

Distributed representations allow the topics "government", "corruption" and "oil" to combine to give very high probability to a word "Putin".
Deep Boltzmann Machines

Low-level features:
- Edges

Built from **unlabeled** inputs.

Input: Pixels

(Original cited from Salakhutdinov & Hinton, Neural Computation 2012)
Deep Boltzmann Machines

Learn simpler representations, then compose more complex ones

Input: Pixels

Built from **unlabeled** inputs.

Higher-level features: Combination of edges

Low-level features: Edges

(Dense) Boltzmann Machines

Learn simpler representations, then compose more complex ones

Input: Pixels

Built from **unlabeled** inputs.

(Salakhutdinov 2008, Salakhutdinov & Hinton 2012)
Model Formulation

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

- Dependencies between hidden variables.
- All connections are undirected.

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

- Bottom-up and Top-down:

\[ P(h_j^2 = 1|h^1, h^3) = \sigma \left( \sum_k W_{k,j}^3 h_k^3 + \sum_m W_{m,j}^2 h_m^1 \right) \]

- Hidden variables are dependent even when \textit{conditioned on} the input.
Mathematical Formulation

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

Deep Boltzmann Machine

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

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

Deep Boltzmann Machine

Neural Network

Output

Deep Belief Network

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

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

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

- Dependencies between hidden variables.

Maximum likelihood learning:

\[ \frac{\partial \log P_\theta(v)}{\partial W^1} = E_{P_{data}}[vh^1^T] - E_{P_\theta}[vh^1^T] \]

**Problem:** Both expectations are intractable!

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

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

(Approximate) Maximum Likelihood:

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

- Both expectations are intractable!

\[ P_{data}(v, h^1) = P_{\theta}(h^1|v)P_{data}(v) \]

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

Not factorial any more!
Approximate Learning

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

(Approximate) Maximum Likelihood:

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

\[ P_{data}(v, h^{(1)}) = P_\theta(h^{(1)}|v)P_{data}(v) \]

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

Not factorial any more!
Approximate Learning

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

(Approximate) Maximum Likelihood:

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

Variational Inference

Stochastic Approximation (MCMC-based)

Not factorial any more!

\[ P_{data}(v, h^{1}) = P_{\theta}(h^{1}|v) P_{data}(v) \]

\[ P_{data}(v) = \frac{1}{N} \sum_{n=1}^{N} \delta(v - v_n) \]
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)
- 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**.

New Learning Algorithm

**Posterior Inference**

- Conditional

Approximate conditional $P_{data}(h|v)$

**Simulate from the Model**

- Approximate the joint distribution $P_{model}(h, v)$

(Salakhutdinov, 2008; NIPS 2009)
New Learning Algorithm

Posterior Inference

Conditional

Approximate conditional

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

Data-dependent

\( E_{P_{data}}[vh^\top] \)

Simulate from the Model

Approximate the joint distribution

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

Data-independent

\( E_{P_{model}}[vh^\top] \)

Match

(Salakhutdinov, 2008; NIPS 2009)
New Learning Algorithm

Posterior Inference

Conditional

Mean-Field

\[ E_{P_{data}}[vh^\top] \]
Data-dependent

\[ E_{P_{model}}[vh^\top] \]
Data-independent

Unconditional

Markov Chain Monte Carlo

Key Idea:

Data-dependent: Variational Inference, mean-field theory

Data-independent: Stochastic Approximation, MCMC based
Stochastic Approximation

Update $\theta_t$ and $x_t$ sequentially, where $x = \{v, h^1, h^2\}$

- Generate $x_t \sim T_{\theta_t}(x_t \leftarrow x_{t-1})$ by simulating from a Markov chain that leaves $P_{\theta_t}$ invariant (e.g. Gibbs or M-H sampler)
- Update $\theta_t$ by replacing intractable $E_{P_{\theta_t}}[vh^\top]$ with a point estimate $\hat{v}_t h_t^\top$

In practice we simulate several Markov chains in parallel.

L. Younes, Probability Theory 1989
Learning Algorithm

Update rule decomposes:

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

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 be any valid transition operator – Tempered Transitions, Parallel/Simulated Tempering.

Connections to the theory of stochastic approximation and adaptive MCMC.

(Salakhutdinov, ICML 2010, NIPS 2011, Srivastava & Salakhutdinov, NIPS 2012, Grosse et.al., 2013, Burda et.al., 2015);
Variational Inference

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

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

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

$$= \sum_h Q_\mu(h|v) \log P_\theta^*(h, v) - \log Z(\theta) + \sum_h Q_\mu(h|v) \log \frac{1}{Q_\mu(h|v)}$$

$$= \log P_\theta(v) - KL(Q_\mu(h|v) || P_\theta(h|v))$$

Minimize $KL$ between approximating and true distributions with respect to variational parameters $\mu$.

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

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

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

Variational Lower Bound

**Mean-Field**: Choose a fully factorized distribution:

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

**Variational Inference**: Maximize the lower bound w.r.t. Variational parameters $\mu$.

Nonlinear fixed-point equations:

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

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

$$\mu_m^{(3)} = \sigma\left(\sum_k W_{km}^3 \mu_k^{(2)}\right)$$
Variational Inference

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

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

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

2. **MCMC**: Apply stochastic approximation to update model parameters

Almost sure convergence guarantees to an asymptotically stable point.

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

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

\[
\log P_{\theta}(v) \geq \log P_{\theta}(v) - \text{KL}(Q_{\mu}(h|v) || P_{\theta}(h|v))
\]

1. Variational Inference: Maximize the lower bound w.r.t. variational parameters.
2. MCMC: Apply stochastic approximation to update model parameters.

Almost sure convergence guarantees to an asymptotically stable point.

Fast Inference

Learning can scale to millions of examples

Unconditional Simulation

Markov Chain Monte Carlo
Good Generative Model?

Handwritten Characters
Good Generative Model?

Handwritten Characters
Good Generative Model?

Handwritten Characters

Simulated          Real Data
Good Generative Model?

Handwritten Characters

Real Data   Simulated
Good Generative Model?

Handwritten Characters
Good Generative Model?

MNIST Handwritten Digit Dataset
Handwriting Recognition

MNIST Dataset
60,000 examples of 10 digits

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic regression</td>
<td>12.0%</td>
</tr>
<tr>
<td>K-NN</td>
<td>3.09%</td>
</tr>
<tr>
<td>Neural Net (Platt 2005)</td>
<td>1.53%</td>
</tr>
<tr>
<td>SVM (Decoste et.al. 2002)</td>
<td>1.40%</td>
</tr>
<tr>
<td>Deep Autoencoder (Bengio et. al. 2007)</td>
<td>1.40%</td>
</tr>
<tr>
<td>Deep Belief Net (Hinton et. al. 2006)</td>
<td>1.20%</td>
</tr>
<tr>
<td>DBM</td>
<td>0.95%</td>
</tr>
</tbody>
</table>

Optical Character Recognition
42,152 examples of 26 English letters

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic regression</td>
<td>22.14%</td>
</tr>
<tr>
<td>K-NN</td>
<td>18.92%</td>
</tr>
<tr>
<td>Neural Net</td>
<td>14.62%</td>
</tr>
<tr>
<td>SVM (Larochelle et.al. 2009)</td>
<td>9.70%</td>
</tr>
<tr>
<td>Deep Autoencoder (Bengio et. al. 2007)</td>
<td>10.05%</td>
</tr>
<tr>
<td>Deep Belief Net (Larochelle et. al. 2009)</td>
<td>9.68%</td>
</tr>
<tr>
<td>DBM</td>
<td>8.40%</td>
</tr>
</tbody>
</table>

Permutation-invariant version.
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

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic regression</td>
<td>22.5%</td>
</tr>
<tr>
<td>K-NN (LeCun 2004)</td>
<td>18.92%</td>
</tr>
<tr>
<td>SVM (Bengio &amp; LeCun 2007)</td>
<td>11.6%</td>
</tr>
<tr>
<td>Deep Belief Net (Nair &amp; Hinton 2009)</td>
<td>9.0%</td>
</tr>
<tr>
<td>DBM</td>
<td>7.2%</td>
</tr>
</tbody>
</table>

Permutation-invariant version.
Talk Roadmap

• Learning Deep Models
  – Restricted Boltzmann Machines
  – Deep Boltzmann Machines

• Multi-Modal Learning with DBMs

Srivastava & Salakhutdinov,
JMLR 2014, NIPS 2012

• Evaluating Deep Generative Models

Nitish Srivastava
Data – Collection of Modalities

• Multimedia content on the web - image + text + audio.

• Product recommendation systems.

• Robotics applications.

Touch sensors

Vision

Audio

Motor control

sunset, pacificocean, bakerbeach, seashore, ocean, car, automobile

Netflix, YouTube, flickr, Google, eBay, amazon
Shared Concept

“Modality-free” representation

“Modality-full” representation

“Concept”

- sunset, pacific ocean, baker beach, seashore, ocean
Multi-Modal Input

• Improve Classification
  pentax, k10d, kangarooisland southaustralia, sa australia australiansealion 300mm
  SEA / NOT SEA

• Fill in Missing Modalities
  beach, sea, surf, strand, shore, wave, seascape, sand, ocean, waves

• Retrieve data from one modality when queried using data from another modality
  beach, sea, surf, strand, shore, wave, seascape, sand, ocean, waves
Challenges - I

Very different input representations

- Images – real-valued, dense
- Text – discrete, sparse

Difficult to learn cross-modal features from low-level representations.
Challenges - II

Noisy and missing data

Text:
- pentax, k10d, pentaxda50200, kangarooisland, sa, australiansealion
- mickikrimmel, mickipedia, headshot
- < no text>
- unseulpixel, naturey, crap
# Challenges - II

<table>
<thead>
<tr>
<th>Image</th>
<th>Text</th>
<th>Text generated by the model</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Image](pentax, k10d, pentaxda50200, kangarooisland, sa, australiansealion)</td>
<td>pentax, k10d, pentaxda50200, kangarooisland, sa, australiansealion</td>
<td>beach, sea, surf, strand, shore, wave, seascape, sand, ocean, waves</td>
</tr>
<tr>
<td>![Image](mickikrimmel, mickipedia, headshot)</td>
<td>mickikrimmel, mickipedia, headshot</td>
<td>portrait, girl, woman, lady, blonde, pretty, gorgeous, expression, model</td>
</tr>
<tr>
<td><img src="%20no%20text" alt="Image" /></td>
<td>&lt; no text&gt;</td>
<td>night, notte, traffic, light, lights, parking, darkness, lowlight, nacht, glow</td>
</tr>
<tr>
<td>![Image](unseulpixel, naturey, crap)</td>
<td>unseulpixel, naturey, crap</td>
<td>fall, autumn, trees, leaves, foliage, forest, woods, branches, path</td>
</tr>
</tbody>
</table>
A Simple Multimodal Model

- Use a joint binary hidden layer.

**Problem**: Inputs have very different statistical properties.

- Difficult to learn cross-modal features.
Multimodal DBM

Dense, real-valued image features

Gaussian model

Replicated Softmax

Word counts

(Srivastava & Salakhutdinov, NIPS 2012)
Multimodal DBM

Gaussian model

Dense, real-valued image features

$\mathbf{v}_{\text{image}}$

Replicated Softmax

Word counts

(Srivastava & Salakhutdinov, NIPS 2012)
Multimodal DBM

Gaussian model

Dense, real-valued image features

\( \mathbf{V}_{\text{image}} \)

\( \mathbf{V}_{\text{text}} \) (Srivastava & Salakhutdinov, NIPS 2012)

Replicated Softmax

Word counts
Multimodal DBM

Bottom-up
+
Top-down

Gaussian model

Dense, real-valued image features

V_{image}

Replicated Softmax

Word counts

V_{text}
Multimodal DBM

\[
P(v^m, v^t; \theta) = \sum_{h^{(2m)}, h^{(2t)}, h^{(3)}} P(h^{(2m)}, h^{(2t)}, h^{(3)}) \left( \sum_{h^{(1m)}} P(v_m, h^{(1m)}|h^{(2m)}) \right) \left( \sum_{h^{(1t)}} P(v^t, h^{(1t)}|h^{(2t)}) \right)
\]

\[
\frac{1}{Z(\theta, M)} \sum_{h} \exp \left( -\sum_i \frac{(v_i^m)^2}{2\sigma_i^2} + \sum_{ij} \frac{v_i^m}{\sigma_i} w_{ij}^{(1m)} h_j^{(1m)} + \sum_{jl} w_{jl}^{(2m)} h_j^{(1m)} h_l^{(2m)} \right)
\]

Gaussian Image Pathway

\[
+ \sum_{jk} w_{kj}^{(1t)} h_j v_k^t + \sum_{jl} w_{jl}^{(2t)} h_j^{(1t)} h_l^{(2t)} + \sum_{lp} w_{lp}^{(3t)} h_l^{(2t)} h_p^{(3)} + \sum_{lp} w_{lp}^{(3m)} h_l^{(2m)} h_p^{(3)}
\]

Replicated Softmax Text Pathway

Joint 3rd Layer

V_{image} V_{text}
Text Generated from Images

Given: dog, cat, pet, kitten, puppy, ginger, tongue, kitty, dogs, furry

Generated: sea, france, boat, mer, beach, river, bretagne, plage, brittany

Given: portrait, child, kid, ritratto, kids, children, boy, cute, boys, italy

Generated: insect, butterfly, insects, bug, butterflies, lepidoptera

Given: graffiti, streetart, stencil, sticker, urbanart, graff, sanfrancisco

Generated: canada, nature, sunrise, ontario, fog, mist, bc, morning
Given

- portrait, women, army, soldier, mother, postcard, soldiers

Generated

- obama, barackobama, election, politics, president, hope, change, sanfrancisco, convention, rally

- water, glass, beer, bottle, drink, wine, bubbles, splash, drops, drop
Images from Text

**Given**
- water, red, sunset
- nature, flower, red, green
- blue, green, yellow, colors
- chocolate, cake

**Retrieved**
MIR-Flickr Dataset

• 1 million images along with user-assigned tags.

Huiskes et. al.
Data and Architecture

\[ \approx 12 \text{ Million parameters} \]

- 200 most frequent tags.
- 25K labeled subset (15K training, 10K testing)
- Additional 1 million unlabeled data
- 38 classes - sky, tree, baby, car, cloud ...
Results

• Logistic regression on top-level representation.

• Multimodal Inputs

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>MAP</th>
<th>Precision@50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.124</td>
<td>0.124</td>
</tr>
<tr>
<td>LDA [Huiskes et. al.]</td>
<td>0.492</td>
<td>0.754</td>
</tr>
<tr>
<td>SVM [Huiskes et. al.]</td>
<td>0.475</td>
<td>0.758</td>
</tr>
<tr>
<td>DBM-Labelled</td>
<td>0.526</td>
<td>0.791</td>
</tr>
</tbody>
</table>

Mean Average Precision

Labeled 25K examples
Results

• Logistic regression on top-level representation.

• Multimodal Inputs

<table>
<thead>
<tr>
<th>Learning Algorithm</th>
<th>MAP</th>
<th>Precision@50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.124</td>
<td>0.124</td>
</tr>
<tr>
<td>LDA [Huiskes et. al.]</td>
<td>0.492</td>
<td>0.754</td>
</tr>
<tr>
<td>SVM [Huiskes et. al.]</td>
<td>0.475</td>
<td>0.758</td>
</tr>
<tr>
<td>DBM-Labelled</td>
<td>0.526</td>
<td>0.791</td>
</tr>
<tr>
<td>Deep Belief Net</td>
<td>0.638</td>
<td>0.867</td>
</tr>
<tr>
<td>Autoencoder</td>
<td>0.638</td>
<td>0.875</td>
</tr>
<tr>
<td>DBM</td>
<td>0.641</td>
<td>0.873</td>
</tr>
</tbody>
</table>

Mean Average Precision

Labeled 25K examples + 1 Million unlabelled
Generating Sentences

• More challenging problem.
• How can we generate complete descriptions of images?

Input

Output

A man skiing down the snow covered mountain with a dark sky in the background.

Second Half of Tutorial
Talk Roadmap

• Learning Deep Models
  – Restricted Boltzmann Machines
  – Deep Boltzmann Machines

• Multi-Modal Learning with DBMs

• Evaluating Deep Generative Models

Burda, Grosse, Salakhutdinov, AI & Statistics 2015

Yura Burda  Roger Grosse
Markov Random Fields

**Graphical Models:** Powerful framework for representing dependency structure between random variables.

\[
P_\theta(x) = \frac{1}{\mathcal{Z}(\theta)} \exp(-E(x; \theta)) = \frac{f_\theta(x)}{\mathcal{Z}(\theta)}
\]

Partition function: difficult to compute

\[
\mathcal{Z}(\theta) = \sum_x \exp(-E(x; \theta))
\]

- **Goal:** Obtain good estimates of \( \mathcal{Z}(\theta) \).
Restricted Boltzmann Machines

Stochastic binary visible variables \( v \in \{0, 1\}^D \) are connected to stochastic binary hidden variables \( h \in \{0, 1\}^F \).

The energy of the joint configuration:

\[
E(v, h; \theta) = - \sum_{i,j} W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j a_j h_j
\]

\( \theta = \{W, a, b\} \) model parameters.

Probability of the joint configuration is given by the Boltzmann distribution:

\[
P_{\theta}(v) = \frac{1}{\mathcal{Z}(\theta)} \sum_h \exp(-E(v, h; \theta)) = \frac{f_{\theta}(v)}{\mathcal{Z}(\theta)}.
\]

Markov random fields, Boltzmann machines, log-linear models.
Model Selection

• Model Selection / Complexity Control?

• Suppose we have two MRFs with parameters $\theta_A$ and $\theta_B$.

• Each MRF has different number of hidden units and was trained using different learning rates and different numbers of CD steps.

• On the validation set, we need to compute:

$$\frac{P(v; \theta_A)}{P(v; \theta_B)} = \frac{f_{\theta_A}(v)}{f_{\theta_B}(v)} \times \frac{Z(\theta_B)}{Z(\theta_A)}.$$ 

• This requires knowing the ratio of partition functions.
Generative Model

- Which model is a better generative model?

Model A

Model B
Model Selection

• More generally, how can we choose between models?

RBM samples

Mixture of Bernoulli’s

Compare $P(x)$ on the validation set: $P(x) = f(x)/\mathcal{Z}$.

Need an estimate of Partition Function $\mathcal{Z}$
Model Selection

• More generally, how can we choose between models?

RBM samples
MoB, test log-probability: -137.64 nats/digit
RBM, test log-probability: -86.35 nats/digit

Difference of about 50 nats!
Simple Importance Sampling

- Two distributions defined on $\mathcal{X}$ with probability distribution functions $p_{\text{ini}}(x) = \frac{f_{\text{ini}}(x)}{Z_0}$ and $p_{\text{tgt}}(x) = \frac{f_{\text{tgt}}(x)}{Z_{\text{tgt}}}$
  - Proposal, easy to sample from distribution
  - Intractable, target distribution

- Under mild conditions:
  \[ Z_{\text{tgt}} = \sum_x f_{\text{tgt}}(x) = \sum_x \frac{f_{\text{tgt}}(x)}{p_{\text{ini}}(x)} \times p_{\text{ini}}(x) \]

- Get an unbiased estimate by using Monte Carlo approximation:
  \[ Z_{\text{tgt}} \approx \frac{1}{M} \sum_{m=1}^{M} f_{\text{tgt}}(x^{(m)}) = \frac{1}{M} \sum_{m=1}^{M} w^{(m)} \quad x^{(m)} \sim p_{\text{ini}} \]

- In high-dimensional spaces, the variance will be high (or infinite).
Annealing Between Distributions

• Consider a sequence of intermediate distributions: \( p_0, p_1, \ldots, p_K \) with \( p_0 = p_{\text{ini}} \) and \( p_K = p_{\text{tgt}} \).

• One general way is to use geometric averages:

\[
p_\beta(x) = \frac{f_\beta(x)}{Z_\beta} = \frac{f_{\text{ini}}(x)^{1-\beta} f_{\text{tgt}}(x)^\beta}{Z_\beta}
\]

with \( 0 = \beta_0 < \beta_1 < \ldots < \beta_K = 1 \) chosen by the user.

• If \( p_{\text{ini}} \) is the uniform distribution, then:

\[
p_\beta(x) = \frac{f_{\text{tgt}}(x)^\beta}{Z_\beta}
\]

hence the term annealing.
Annealing Between Distributions

- Move gradually from hotter distribution to colder distribution:

- Need to define transition operator $T_k(x'|x)$ that leaves $p_k$ invariant (e.g. Gibbs sampling) – Easy to implement!
Annealed Importance Sampling Run

\[ x_1 \sim T_1(x|x_0) \quad \cdots \quad x_{K-1} \sim T_{K-1}(x|x_{K-2}) \]

\[ x_0 \sim p_0(x) \]

- Generate: \( x_0, x_1, \ldots, x_{K-1} \)
  - Sample \( x_0 \sim p_0(x) \)
  - Sample \( x_1 \sim T_1(x|x_0) \)
  - ...
  - Sample \( x_{K-1} \sim T_{K-1}(x|x_{K-2}) \)

\[
 w^{(m)} = \frac{f_{tgt}(x_{K-1}) f_1(x_0) f_2(x_1) \cdots f_{K-1}(x_{K-2})}{p_0(x_0) f_1(x_1) f_2(x_2) \cdots f_{K-1}(x_{K-1})}
\]

- We obtain an unbiased estimator: \( \mathbb{E}[w] = \mathcal{Z}_{tgt} \)
AIS is Importance Sampling

\[
x_1 \sim T_1(x|x_0)
\]

\[
x_{K-1} \sim T_{K-1}(x|x_{K-2})
\]

\[
x_0 \sim p_0(x)
\]

\[
x_0 \sim \tilde{T}_1(x|x_1)
\]

\[
x_{K-1} \sim p_{tgt}(x)
\]

- Forward Markov chain:
  \[
  q_{fwd}(x_0, \ldots, x_{K-1}) = p_0(x_0) \prod_{k=1}^{K-1} T_k(x_k|x_{k-1})
  \]

- Reverse Markov chain (merely a theoretical construct):
  \[
  f_{rev}(x_0, \ldots, x_{K-1}) = f_{tgt}(x_{K-1}) \prod_{k=1}^{K-1} \tilde{T}_k(x_{k-1}|x_k)
  \]

  \[
  \tilde{T}_k(x' | x) = T_k(x | x') p_k(x') / p_k(x)
  \]
AIS is Importance Sampling

\[ x_1 \sim T_1(x|x_0) \]

\[ x_0 \sim p_0(x) \]

\[ x_0 \sim \tilde{T}_1(x|x_1) \]

\[ \ldots \]

\[ x_{K-1} \sim T_{K-1}(x|x_{K-2}) \]

\[ x_{K-1} \sim p_{tgt}(x) \]

- AIS is a simple importance sampling on extended space:

\[
\mathcal{Z}_{tgt} = \mathbb{E}_{q_{fwd}} \left[ \frac{f_{\text{rev}}}{q_{fwd}} \right] = \mathbb{E}_{q_{fwd}}[\mathcal{W}]
\]
RBMs with Geometric Averages

- Restricted Boltzmann Machines trained on MNIST.

Samples from target distribution

AIS with geometric averages

beta = 0.00
Problems with Undirected Models

• AIS provides an unbiased estimator: \( \mathbb{E}[\hat{Z}_{tgt}] = Z_{tgt} \). In general, we are interested in estimating \( \log Z_{tgt} \)

• By Jensen’s inequality:

\[
\mathbb{E}[\log \hat{Z}_{tgt}] \leq \log \mathbb{E}[\hat{Z}_{tgt}] = \log Z_{tgt}
\]

• By Markov’s inequality: very unlikely to overestimate \( \log Z_{tgt} \)

\[
\Pr(\log \hat{Z}_{tgt} > \log Z_{tgt} + b) \leq e^{-b}
\]

Stochastic lower bound!

• Compute log-probability on the test set:

\[
\log p(x) = \log f(x) - \log Z_{tgt}
\]

overestimate underestimate
Motivation: RBM Sampling

Run Markov chain (alternating Gibbs Sampling):
Motivation: RBM Sampling

Run Markov chain (alternating Gibbs Sampling):

\[ \n \vax
\]
Motivation: RBM Sampling

Run Markov chain (alternating Gibbs Sampling):

\[
P(h | v)
\]

\[
h \quad \bullet \quad \bullet \quad \bullet
\]

\[
v \quad \bullet \quad \bullet \quad \bullet \quad \bullet
\]

Random

\[
P(h | v) = \prod_j P(h_j | v) \quad P(h_j = 1 | v) = \frac{1}{1 + \exp(- \sum_i W_{ij} v_i - a_j)}
\]
Motivation: RBM Sampling

Run Markov chain (alternating Gibbs Sampling):

\[
P(h|v)
\]

Random

V

\[
P(h|v) = \prod_j P(h_j|v) \quad P(h_j = 1|v) = \frac{1}{1 + \exp(-\sum_i W_{ij}v_i - a_j)}
\]

\[
P(v|h) = \prod_i P(v_i|h) \quad P(v_i = 1|h) = \frac{1}{1 + \exp(-\sum_j W_{ij}h_j - b_i)}
\]
Motivation: RBM Sampling

Run Markov chain (alternating Gibbs Sampling):

\[ P(h|v) \]

\[ P(h_j = 1|v) = \frac{1}{1 + \exp(-\sum_i W_{ij}v_i - a_j)} \]

\[ P(v|h) = \prod_i P(v_i|h) \]

\[ P(v_i = 1|h) = \frac{1}{1 + \exp(-\sum_j W_{ij}h_j - b_i)} \]
Motivation: RBM Sampling

Run Markov chain (alternating Gibbs Sampling):

\[ P(h|v) \]

1 Gibbs step: Transition operator \( T \).

Equilibrium Distribution

\[
P(h|v) = \prod_j P(h_j|v) \quad P(h_j = 1|v) = \frac{1}{1 + \exp(-\sum_i W_{ij}v_i - a_j)}
\]

\[
P(v|h) = \prod_i P(v_i|h) \quad P(v_i = 1|h) = \frac{1}{1 + \exp(-\sum_j W_{ij}h_j - b_i)}
\]
Motivation: RBM Sampling

Run Markov chain (alternating Gibbs Sampling):

\[ P(h|v) \]

\[ P(h_j = 1|v) = \frac{1}{1 + \exp(-\sum_i W_{ij}v_i - a_j)} \]

\[ P(v|h) \]

\[ P(v_i = 1|h) = \frac{1}{1 + \exp(-\sum_j W_{ij}h_j - b_i)} \]
Unrolled RBM as a Deep Generative Model

Random (uniform)

$\begin{array}{c}
\text{ uniform } \\
v
\end{array}$
Unrolled RBM as a Deep Generative Model

Random (uniform)

\[
\begin{align*}
\begin{array}{c}
\text{O} \text{O} \text{O} \text{O} \\
\text{v} \\
\downarrow \\
\text{W}^T \\
\ldots
\end{array}
\end{align*}
\]
Unrolled RBM as a Deep Generative Model

Random (uniform)

\[ \begin{array}{c}
\circ \circ \circ \ v \\
\downarrow \ W^T \\
\cdots \\
\circ \circ \circ \ v
\end{array} \]
Unrolled RBM as a Deep Generative Model

Random (uniform)

\[
\begin{array}{c}
\text{O} \quad \text{O} \quad \text{O} \quad \text{O} \\
\downarrow \quad W^T \\
\text{...} \\
\text{O} \quad \text{O} \quad \text{O} \quad \text{O} \\
\downarrow \quad W^T \\
\text{O} \quad \text{O} \quad h
\end{array}
\]
Unrolled RBM as a Deep Generative Model

Random (uniform)

\[
\begin{align*}
&\text{Observed Data} \\
&v \\
&\downarrow^{W^T} \\
&\cdots \\
&v \\
&\downarrow^{W^T} \\
&h \\
&\downarrow W \\
&v
\end{align*}
\]
Unrolled RBM as a Deep Generative Model

Random (uniform)

\[
\begin{align*}
\mathbf{v} & \quad W^\top \quad \mathbf{W}^\top \quad \mathbf{h} \quad W \\
\mathbf{v} & \quad W^\top \quad \mathbf{W}^\top \quad \mathbf{h} \\
\mathbf{v} & \quad W \quad \mathbf{W} \\
\text{Observed Data}
\end{align*}
\]

• If we use infinite number of layers, then:

\[ P_{gen}(\mathbf{v}) = P_{RBM}(\mathbf{v}) \]

• Otherwise, deep generative model is just an approximation to an RBM.
Reverse AIS Estimator (RAISE)

Let us consider \( \mathbf{x} = \{ \mathbf{v}, \mathbf{h} \} \) where \( \mathbf{v} \) is observed and \( \mathbf{h} \) is unobserved.

Define the following generative process (sequence of AIS distributions):

\[
p_{\text{fwd}}(\mathbf{x}_{0:K}) = p_0(\mathbf{x}_0) \prod_{k=1}^{K} T_k(\mathbf{x}_k|\mathbf{x}_{k-1})
\]

Generative model, that we call the annealing model:

\[
p_{\text{ann}}(\mathbf{v}_K) = \sum_{\mathbf{x}_{0:K-1}, \mathbf{h}_K} p_{\text{fwd}}(\mathbf{x}_{0:K-1}, \mathbf{h}_K, \mathbf{v}_K)
\]
Reverse AIS Estimator (RAISE)

\[ P_{\text{ann}}(x) = P_{\text{RBM}}(x) \]

As \( K \) goes to infinity:

We would like to estimate \( p(v_{\text{test}}) \).

We use reverse chain as our proposal:

\[ q_{\text{rev}}(x_{0:K-1}, h_K | v_{\text{test}}) = \]

\[ p_{\text{tgt}}(h_K | v_{\text{test}}) \prod_{k=1}^{K} \tilde{T}_k(x_{k-1} | x_k) \]

Assume tractable, which is the case for RBMs

Can be easily extended to non-tractable posteriors, e.g. DBMs, DBNs.
Reverse AIS Estimator (RAISE)

• We now have our generative model (theoretical construct):

\[ p_{\text{fwd}}(x_{0:K}) = p_0(x_0) \prod_{k=1}^{K} T_k(x_k | x_{k-1}) \]

• Proposal starts at the data and melts the distribution:

\[ q_{\text{rev}}(x_{0:K-1}, h_K | v_{\text{test}}) = p_{\text{tgt}}(h_K | v_{\text{test}}) \prod_{k=1}^{K} \tilde{T}_k(x_{k-1} | x_k) \]

• We then obtain:

\[ P_{\text{ann}}(v_{\text{test}}) = \mathbb{E}_{q_{\text{rev}}} \left[ \frac{f_{\text{fwd}}}{q_{\text{rev}}} \right] = \mathbb{E}_{q_{\text{rev}}} \left[ \frac{f_{\text{tgt}}(v_{\text{test}})}{Z_0} \prod_{k=1}^{K-1} \frac{f_k(x_k)}{f_{k+1}(x_k)} \right] = \mathbb{E}_{q_{\text{rev}}} [w] \]

• Tends to underestimate rather than overestimate log-probs!
MNIST

- RBM with 500 hidden units trained on MNIST with CD1.

- **Initial distribution is uniform:** AIS with geometric averages is off by 20 nats, even when using 100K intermediate distributions!

![](image)
The only different is the choice of initial distribution.
Omniglot Dataset

- RBM with 500 hidden units trained on Omniglot with PCD.
### MNIST and Omniglot Results

<table>
<thead>
<tr>
<th>Model</th>
<th>exact</th>
<th>CSL</th>
<th>RAISE</th>
<th>AIS</th>
<th>uniform</th>
<th>gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>mnistCD1-20</td>
<td>-164.50</td>
<td>-185.74</td>
<td>-165.33</td>
<td>-164.51</td>
<td></td>
<td>0.82</td>
</tr>
<tr>
<td>mnistPCD-20</td>
<td>-150.11</td>
<td>-152.13</td>
<td>-150.58</td>
<td>-150.04</td>
<td></td>
<td>0.54</td>
</tr>
<tr>
<td>mnistCD1-500</td>
<td>—</td>
<td>-566.91</td>
<td>-150.78</td>
<td>-106.52</td>
<td></td>
<td>44.26</td>
</tr>
<tr>
<td>mnistPCD-500</td>
<td>—</td>
<td>-138.76</td>
<td>-101.07</td>
<td>-99.99</td>
<td></td>
<td>1.08</td>
</tr>
<tr>
<td>mnistCD25-500</td>
<td>—</td>
<td>-145.26</td>
<td>-88.51</td>
<td>-86.42</td>
<td></td>
<td>2.09</td>
</tr>
<tr>
<td>omniPCD-1000</td>
<td>—</td>
<td>-144.25</td>
<td>-100.47</td>
<td>-100.45</td>
<td></td>
<td>0.02</td>
</tr>
</tbody>
</table>

- RAISE errs on the side of underestimating the log-likelihood.
- Note that the gap is very small!
- CSL: Conservative Sampling-based Log-likelihood (CSL) estimator of Bengio et. al.

DBMs and DBNs

Deep Boltzmann Machine

Deep Believe Network
Helmholtz Machines

Approximate Inference

\[ Q(h^3|h^2) \]
\[ Q(h^2|h^1) \]
\[ Q(h^1|v) \]

Generative Process

\[ P(h^2, h^3) \]
\[ P(h^1|h^2) \]
\[ P(v|h^1) \]
Conclusions

• RAISE produces accurate, yet conservative, estimates of log-probabilities for RBMs, DBMs, and DBNs.

• Using both RAISE and AIS, one can judge the accuracy of one’s results by measuring the agreement of the two estimators.

• RAISE is simple to implement (same as AIS), so it gives a simple and practical way to evaluate MRF test log-probabilities.

• These ideas serve as a starting point for learning very deep generative models!
End of Part 1