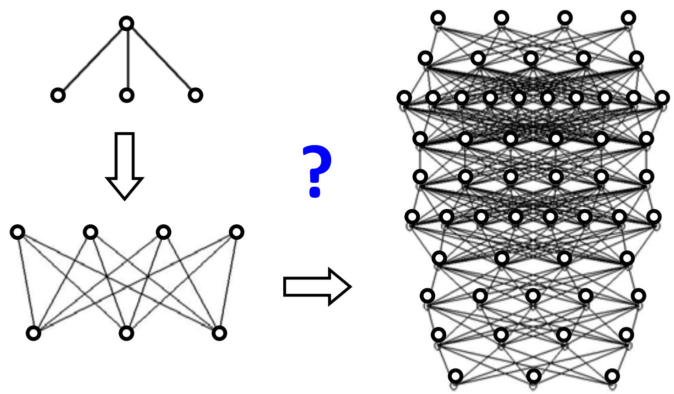
Deep Neural Networks – A Developmental Perspective

B.H. Juang
School of Electrical and Computer Engineering
Georgia Institute of Technology

Stream of Thoughts

- New wave of interest in connectionist’s approach, due to the reported success with Deep Neural Net
- This is a commentary, not about “how to do DNN” but “***how to understand DNN***”
- Follow framework of statistical pattern recognition
  - Understand what is being accomplished and how it is being accomplished
  - Think about what may be missing or otherwise possible

What is Deep Neural Net?



Did it really take 60 years?

Statistical Pattern Recognition

Pattern Recognition – Bayes Theory

Problem Statement:

To identify an unknown observation X as one of M classes (of events or species) with minimum ***probability of error:***

- **Conditional Error**: given X, the cost associated with deciding that it is an i^{th} class event

$$R(i \mid X) = \sum_{j=1}^{M} e_{ij} P(j \mid X)$$

- **Expected error, risk or cost**:

$$C : X \rightarrow i \qquad \mathcal{E} = \int R(C(X) \mid X) p(X) dX$$

How do we design $C(X)$?

Statistical Pattern Recognition

Given a set of design samples $\{X_i, c_i\}_{i=1}^{N}$ with known class identity, where c_i is the class label of X_i, $c_i \in \mathfrak{N}_1^M$, estimate

$$P(j \mid X), \quad j = 1, 2, \cdots, M$$

or

$$P(X \mid j) \text{ and } P(j), \quad j = 1, 2, \cdots, M$$

to implement the Maximum a Posteriori (for 0-1 error) decision to achieve Bayes minimum error.

Essence of statistical methods (vs. heuristics/others):

- ***Learning (finding the distributions) from data***
- "Consistency" with formulation of error probability

Issues in Practice

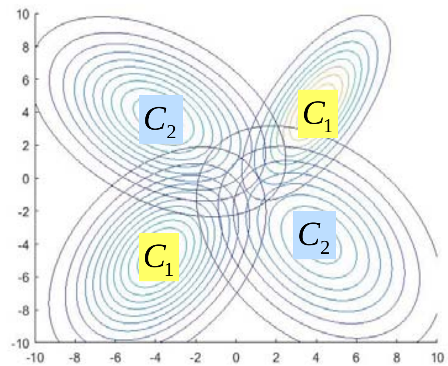
- Raw observations/measurements vs. feature
- Distribution
  - Form: form is in fact substance; form is unknown; wrong form means sub-optimality; form comes before parameter

$$P(X \mid j) \Leftrightarrow P(X \mid j, \theta) \Leftrightarrow P(X \mid j, \tilde{\theta}) \Leftrightarrow \tilde{P}(X \mid j, \tilde{\theta})$$

- Data
  - Quantity: the more the better
  - Quality: sampling **bias** and **error**, rarely explicitly addressed
- Boundary Representation
  - Implicit: let the distribution parameters decide
  - Explicit: define boundary from data w/o assuming distribution

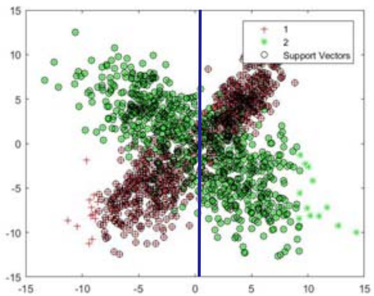
Toy Example – Simulation

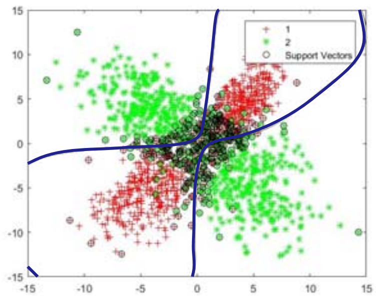
- 2-class problem
- Contour of true pdfs as plotted
- 1000 random points from each class; ~1/4 used for test
- 600 runs



Toy Example – Linear and RBF SVM

- Inseparable 2-class problem
- 1000 random points from each class; ~1/4 used for test
- 600 runs





Average Error Rate (over 600 Runs)

Model		# param	Mean Error rate
Original pdf		20	0.0950
Max of mix pdf		20	0.0958
k-NN (k=1)		1500	0.1316
SVM-linear	Train	2 (~1200)	0.3784
	Test	2	0.3893
SVM-RBF	Train	~500	0.0955
	Test	~500	0.0988
Max mix model pdf		20	0.0965
Mix-model pdf		20	0.0960

SVM with linear boundary fails miserably due to mismatch in the implicit choice of model

Some Important Insights

- System using true distributions is best in both performance and efficiency as predicted by theory
- Decision based on *local* likelihood is competitive in the example – idea of "ensemble of local models"?
- SVM relies on **local optimization**; performs well but is not efficient; may be sensitive to bias
- kNN uses strictly local knowledge but is both un-optimized and un-structured (un-optimized SVM?)
- **Model based recognizer can be as good as any**

ML/PR Problem & Approaches

Need $P(X,j) \rightarrow$ Estimate $P(X \mid j)$ and $P(j)$

$$\Rightarrow P(j \mid X) = P(X \mid j)P(j)/P(X)$$

$P(X \mid j) \leftarrow \tilde{P}(X \mid j, \tilde{\theta})$ Pick a distribution parameterized by θ to match true $P(X \mid j)$

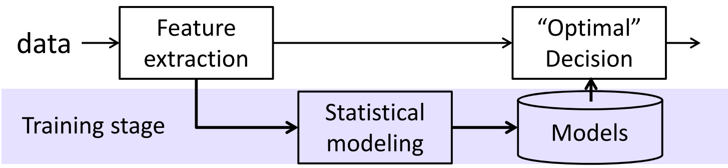
$\leftarrow \tilde{P}(X \mid j, \theta)$ Approximate the true distribution by mixture of easy distributions, e.g., Gaussian

$\leftarrow \tilde{P}(f(X) \mid j, \theta)$ find transform of X to match easy distributions, e.g., Gaussian or MRF

$\leftarrow \tilde{P}(f_{>}(X) \mid j, \theta)$ find transform of X with reduced dimensionality to match easy distributions, e.g., Gaussian or MRF, and possibly mixture of them

“Feature” – Convention & Question

- **Raw data contains components (interference, noise or superfluity) that hinder the decision process**
- Use independent knowledge (*from experts or heuristics*) to extract “feature” from data

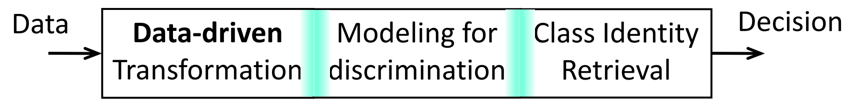


- **Is it possible to accomplish “feature extraction” and “statistical modeling” together “intimately”?**
- **Can we let the data speak for itself ?? !**

Do Not Separate Feature from Model

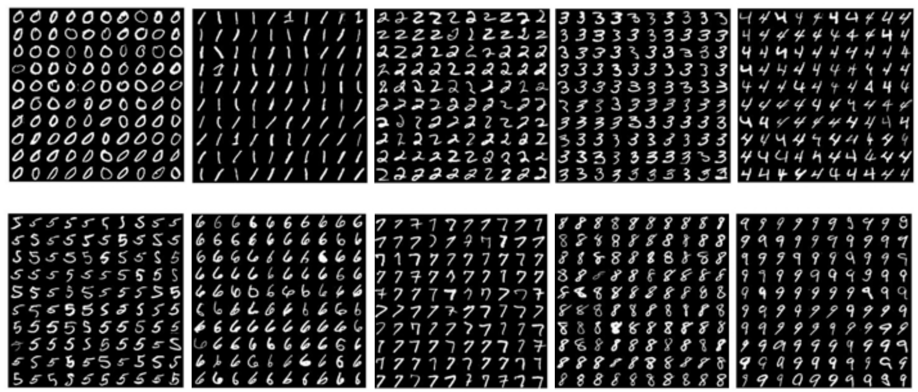
- There is a (bad) tendency to treat the two separately
  - Feature needs to make consistent sense with existing knowledge (so-called experts tell us so)
  - Distribution model is just a tool – pick one you have code to estimate and compute; too casual
- The best feature is one that both makes sense and can be well modeled by a function you can handle

Alternative (and New) Paradigm – fuzzy boundaries between stages



Multivariate Models

NIST Digits



The issue of representation for statistical pattern recognition is finding feature, the uncertainty of which can be characterized by a distribution

How to Represent & Model A Bit Pattern?

Examples:

28  28

$$D^2 = 28 \times 28 = 784 = N$$

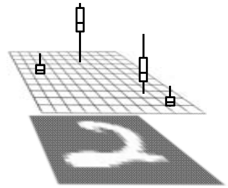
- An D² dimensional binary vector in run-length representation: 
- 28-dimensional vector (each row or column of 28 bits converted to a real number) in Euclidean space – 28-d Gaussian w/ mixture
- An D²-dimensional Gaussian Multivariate

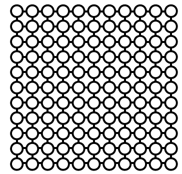
$$p_X(\mathbf{x}) = (2\pi)^{-N/2}\left|\mathbf{C}^{-1}\right|^{1/2}\exp\left\{-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

- Markov Random Field

$$P(X=\mathbf{x}) = \frac{1}{Z}\exp\left(-\sum_i \varphi_i(\mathbf{x})\right) \qquad Z = \sum_x \exp\left(-\sum_i \varphi_i(\mathbf{x})\right)$$

Markov Random Field Model





Neighborhood clique

$$G = (\mathcal{V}, \mathcal{E}) \qquad \mathcal{V} = \{v_i\}_{i=1}^N$$

$$\mathcal{E} = \{(m,n), v_m, v_n \in \mathcal{V}\}$$

$$V_i = \{v_j, (i,j) \in \mathcal{E}, i \neq j\}$$

V_i is clique of node v_i

$\mathcal{C} = \{V_i\}$: clique system

Each node v_i is associated with an r. v. x_i

$$p(x_i \mid x_j, v_j \in (V - v_i)) = p(x_i \mid x_j, v_j \in V_i)$$

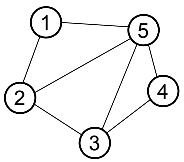
$$p(X=\mathbf{x}) = \frac{1}{Z}\exp\left(-\sum_i \varphi_i(\mathbf{x})\right)$$

A Gibbs measure

Example:

$$\varphi_i(\mathbf{x}) = \sum_{k, v_k \in V_i} w_{k,i} f_{k,i}(x_k)$$

Example of A Clique System



Example for illustration:

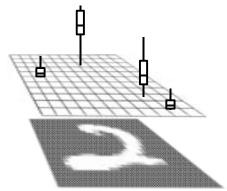
$$\varphi_i(\mathbf{x}) = \sum_{k, v_k \in V_i} w_{k,i} f_{k,i}(x_k)$$

$$\varphi_1(\mathbf{x}) = w_{11} f_1(x_1) + w_{12} f_{12}(x_2) + w_{15} f_{15}(x_5)$$

$$\varphi_5(\mathbf{x}) = w_{55} f_5(x_5) + w_{51} f_{51}(x_1) + w_{52} f_{52}(x_2) + w_{53} f_{53}(x_3) + w_{54} f_{54}(x_4)$$

- System needs not have only uniform connections
- Sparsity in connection can be reflected in the connection weight – in learning, let the data speak!

Gauss-Markov Random Field



MRF $\quad p(X=\mathbf{x}) = \frac{1}{Z}\exp\left(-\sum_i \varphi_i(\mathbf{x})\right)$

$$\varphi_i(\mathbf{x}) = \sum_{k, v_k \in V_i} w_{k,i} f_{k,i}(x_k)$$

Multivariate Gaussian (784 random variables)

$$p_X(\mathbf{x}) = \frac{\left|\mathbf{C}^{-1}\right|^{1/2}}{(2\pi)^{N/2}}\exp\{-\zeta(\mathbf{x};\boldsymbol{\mu},\mathbf{C})\}$$

$$\zeta(\mathbf{x};\boldsymbol{\mu},\mathbf{C}) = \frac{(\mathbf{x}-\boldsymbol{\mu})^t\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})}{2} \Rightarrow \zeta(\mathbf{x};\mathbf{C}) = \frac{\mathbf{x}^t\mathbf{C}^{-1}\mathbf{x}}{2}$$

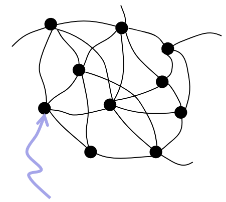
0-mean for simplicity

$$\zeta(\mathbf{x};\mathbf{C}) = r\left(\sum_i a_i x_i^2 + \sum_{i,j\neq i} w_{ij} x_i x_j\right)$$

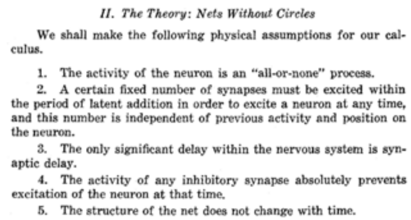
Weights are related to **precision matrix**; clique system imposes constraints on summation, reduces model complexity

Connectionist's Models
– a.k.a. Artificial Neural Networks

Conceptual Neural Networks in Brain



McCulloch-Pitts Neurons



II. The Theory: Nets Without Circles

We shall make the following physical assumptions for our calculus.

1. The activity of the neuron is an "all-or-none" process.
2. A certain fixed number of synapses must be excited within the period of latent addition in order to excite a neuron at any time, and this number is independent of previous activity and position on the neuron.
3. The only significant delay within the nervous system is synaptic delay.
4. The activity of any inhibitory synapse absolutely prevents excitation of the neuron at that time.
5. The structure of the net does not change with time.

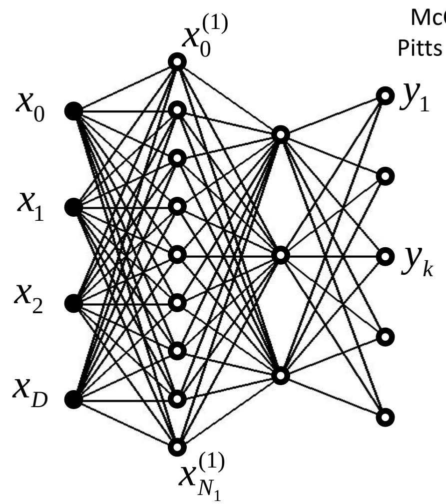
McCulloch-Pitts Neuron, which focuses on computational utility, is one among many models:

- Hodgkin–Huxley
- FitzHugh–Nagumo
- Integrate-and-fire
- Leaky IAF
- Exponential IAF
- Morris–Lecar
- Hindmarsh–Rose
- More …..

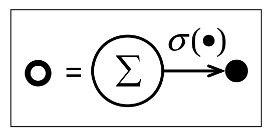
Connectionist Models

- Constitutive processing unit: McCulloch-Pitts Neuron
- **Feedforward Neural Networks (Function Approximation)**
  - Perceptron (Rosenblatt, 1957): single layer feedforward network; kernel perceptron (Aizerman et al, 1974)
  - Multilayer Perceptron or multilayer feedforward neural networks (backpropagation by Werbos, 1974)
- **Recurrent Neural Networks (Auto-association memory, retrieval with partial information)**
  - Hopfield net (Hopfield, 1982); Boltzmann machine (Hinton & Sejnowski, 1985)
  - Self-organizing (feature) map (Kohonen map, 1982) & learning VQ – interpreted as using the closest prototype as retrieved memory

Multilayer Feed-forward Networks



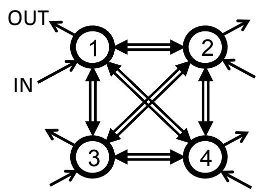
McCulloch-Pitts Neuron



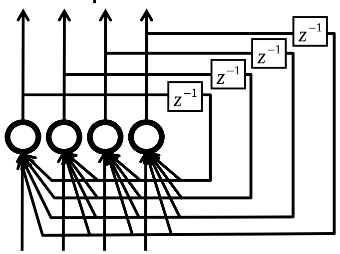
- Number of layers and numbers of hidden units are not constrained;
- D-dimensional input data vector assume real values (non-binary) in most MFNs
- **Temporal** firing activity is usually grossed over
- **Capable of approximating functions w/ arbitrary closeness**

Recurrent Networks – Hopfield Nets

4-node Hopfield Net



Output

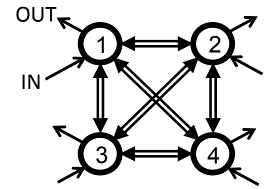


Input

$$E = -\tfrac{1}{2}\sum_{i,j} w_{ij}s_i s_j - \sum_i \tau_i s_i$$

- Hopfield nets are recurrent ANN and serve as content-addressable memory systems with binary threshold nodes; can store ~ $n/(4\log n)$ bit patterns
- System energy is low if the weights emphasize the pair of nodes that behave coherently; given a binary vector input, it converges to one of the "stored" patterns that demonstrates highest coherence

Use of RNN in Decision – No Success



$$\{X_i, c_i\}_{i=1}^{N}$$

$$\mathbf{s} = [\mathbf{x}, c]$$

$$\mathbf{s}_{in} = [\mathbf{x}_{in}, ?] \qquad \mathbf{s}_{out} = [\mathbf{x}_{out}, \tilde{c}]$$

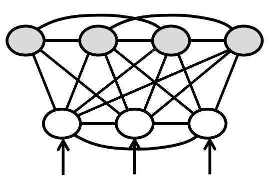
以聯想當決策?　經辨證以決策?

先聯想後辨證?

Boltzmann Machine

A recurrent neural net, similar to Hopfield net, but ***stochastic***

Invisible nodes



Input, visible nodes

Again, binary nodes

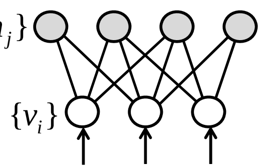
$$E = -\tfrac{1}{2}\sum_{i,j} w_{ij}s_i s_j - \sum_i \tau_i s_i$$

$$\Delta E_i = E_{s_i=0} - E_{s_i=1} = \tfrac{1}{2}\sum_j w_{ij}s_j + \tau_i$$

$$p \approx e^{-\frac{E}{T}} \qquad \frac{p(s_i=0)}{p(s_i=1)} = \frac{1-p_{i1}}{p_{i1}} = e^{-\frac{\Delta E_i}{T}} \rightarrow p_{i1} = \left(1 + e^{-\frac{\Delta E_i}{T}}\right)^{-1}$$

Restricted Boltzmann Machine (RBM)

Invisible nodes



Input, visible nodes

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i,j} w_{ij}v_i h_j - \sum_i a_i v_i - \sum_j b_j h_j$$

$$p(\mathbf{v},\mathbf{h}) = \frac{1}{Z}e^{-E(\mathbf{v},\mathbf{h})}$$

Training Objective

$$\max_{\mathbf{W},\mathbf{a},\mathbf{b}} \prod_{\mathbf{v}\in\mathbf{V}} p(\mathbf{v}) = \max_{\mathbf{W},\mathbf{a},\mathbf{b}} \prod_{\mathbf{v}\in\mathbf{V}} \sum_{\mathbf{h}} p(\mathbf{v},\mathbf{h})$$

$$p(h_j = 1 \mid \mathbf{v}) = \varphi_h\left(b_j + \sum_{i=1}^{M_v} w_{ij}v_i\right); \quad p(v_i = 1 \mid \mathbf{h}) = \varphi_v\left(a_i + \sum_{j=1}^{M_h} w_{ij}h_j\right)$$

- RBM is a generative stochastic RNN: given an input vector it finds **the network state with highest probability**
- **Conditional independence among nodes of same layer**
- Trained with contrastive divergence algorithm

RBM Implements Gauss-Markov R-F

$$p(X=\mathbf{x}) = \frac{1}{Z}\exp\left(-\sum_i \varphi_i(\mathbf{x})\right) \qquad \varphi_i(\mathbf{x}) = \sum_{k,v_k\in V_i} w_{k,i}\, f_{k,i}(x_k)$$

For easy visualization, assume 0-mean,

$$p_X(\mathbf{x}) = \frac{\left|\mathbf{C}^{-1}\right|^{1/2}}{(2\pi)^{N/2}}\exp\{-\zeta(\mathbf{x};\boldsymbol{\mu},\mathbf{C})\} \qquad \zeta(\mathbf{x};\mathbf{0},\mathbf{C}) = r\left(\sum_i a_i x_i^2 + \sum_{i,j\neq i} w_{ij}x_i x_j\right)$$

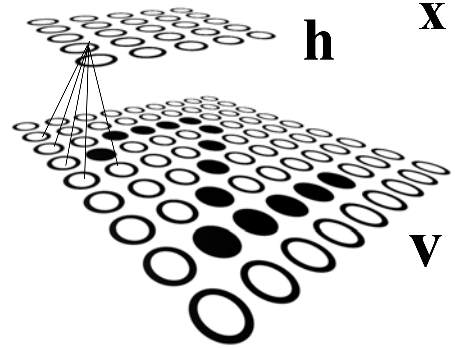
Then for a group of units, treated as binary and hidden

$$\zeta(\mathbf{v},\mathbf{h};\mathbf{0},\mathbf{C}) = r\left(\sum_i a_i x_i^2 + \sum_{i,j\neq i} w_{ij}x_i x_j\right) \approx \sum_i a_i v_i^2 + \sum_j b_j h_j + \sum_{i,j} w_{ij} v_i h_j$$

If all units are binary $\quad \zeta(\mathbf{v},\mathbf{h};\mathbf{0},\mathbf{C}) \approx \sum_i a_i v_i + \sum_j b_j h_j + \sum_{i,j} w_{ij} v_i h_j$

w_{ij} is closely related to elements of the precision matrix

RBM as Vehicle for MRF

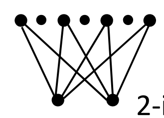


$$\mathbf{x} = [\mathbf{v}, \mathbf{h}]$$

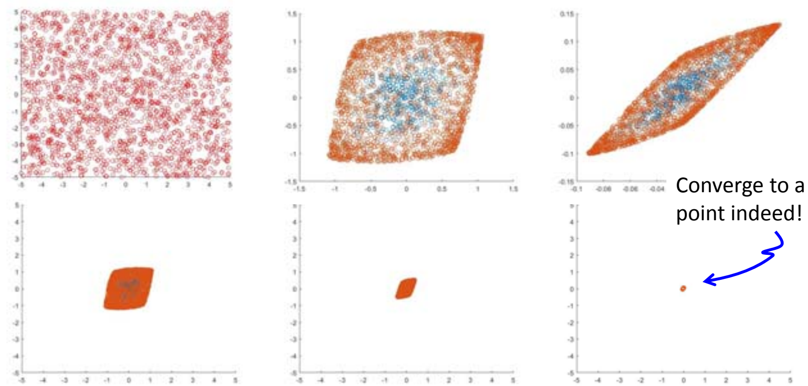
Remarks

- **The Boltzmann machine, as an RNN, converges to most probable state defined by the weight matrix, which has been trained by the provided data**
- Multivariate Gaussian is the underpinning model; pay attention to precision matrix
- Markov property reduces the correlation structure; the clique system needs not be based on adjacency
- RBM learns the multivariate correlation structure of an MRF via the hidden node layer; **it learns the correlation from data, not from human experts**

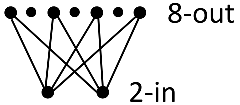
Gaussian-Bernoulli RBM as RNN



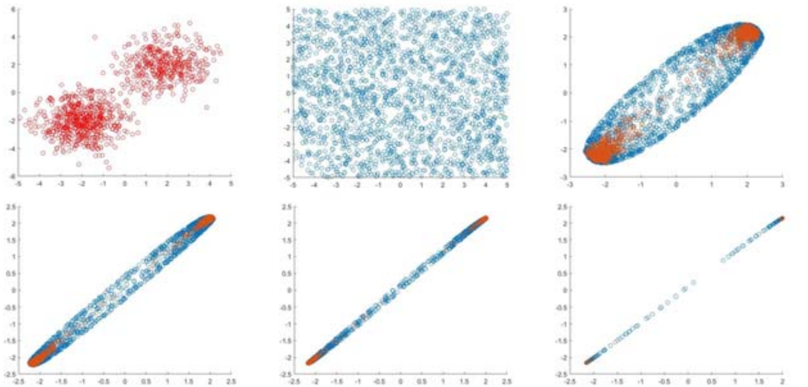
- RBM trained on Gaussian bi-variate – to “memorize” **the location of a point**
- In test, random data in U(-5,5,-5,5)



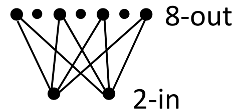
G-B RBM (2Mix 1RBM) – 2-point Memory



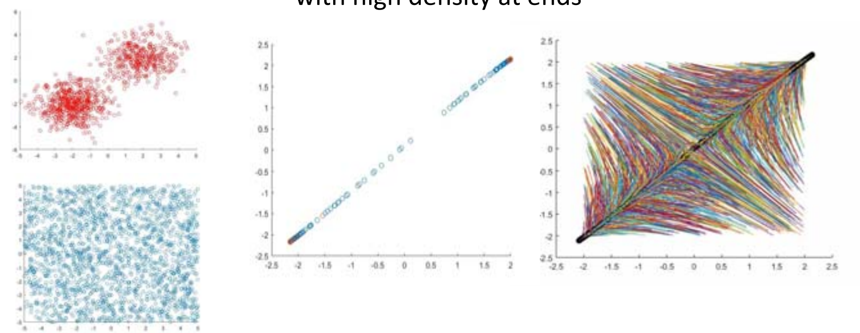
- **Single** RBM trained on 2-mix Gaussian independent bi-variate; tested w/ both 2-mix r.v. and uniform r.v.
- Convergence contour appears a straight line with high density at ends and sparse in-between



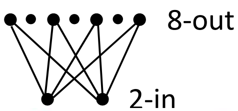
Gaussian-Bernoulli RBM (Attractor)



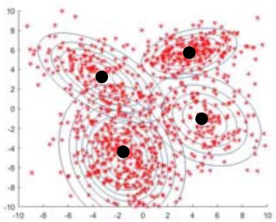
- **Single** RBM trained on 2-mix Gaussian independent bi-variate
- Tested with uniform r.v.
- Convergence contour appears a straight line with high density at ends

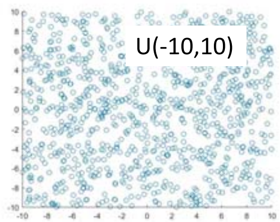


G-B RBM (4-Mix 1-RBM) – 4-pt Memory

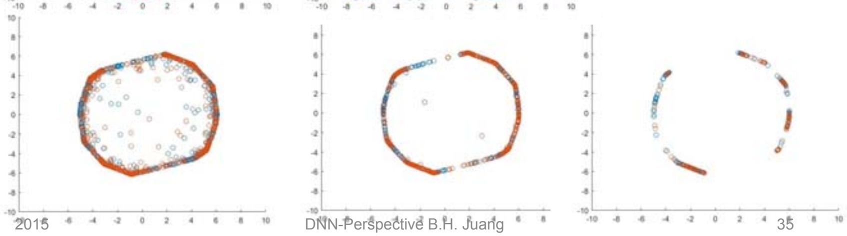


- One RBM trained, on 4-mix bi-variate
- Evaluated on both 4-mix and U(-10,10)





Originally 4 clusters, corresponding to **4-point memory**;

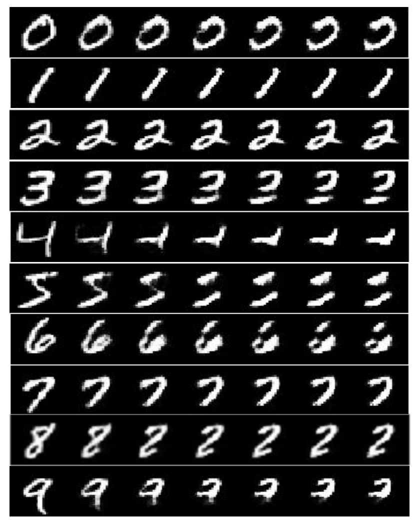


RBM Convergence as an RNN

RBM on "2", 2000 h-units
Input: random bits, "0"-"9"

Iteration →

36 random bit patterns as input



Developmental Remarks

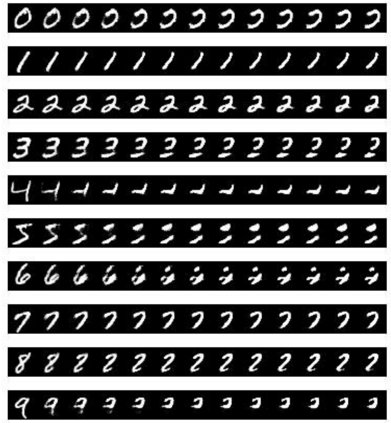
- **The RBM** learns from data the multivariate correlation structure of an MRF via the hidden node layer

28

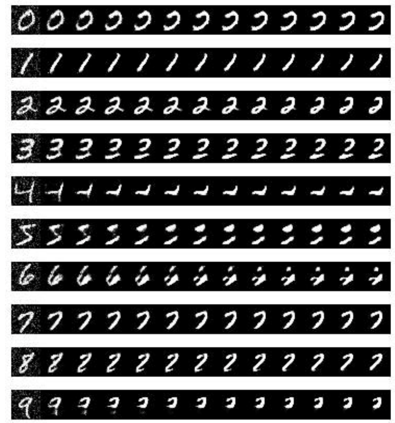
28

- Data dimensionality = 784
- Covariance has 307720 parameters. How many patterns are needed for reliable estimation of the covariance?
- MNIST dataset has ~100K patterns per digit. Is that enough for estimating 307K parameters?
- Casual heuristics do not solve the traditional estimation problem; smart to quickly capture dimension pairs with significant correlation

RBM as RNN with Noisy Input

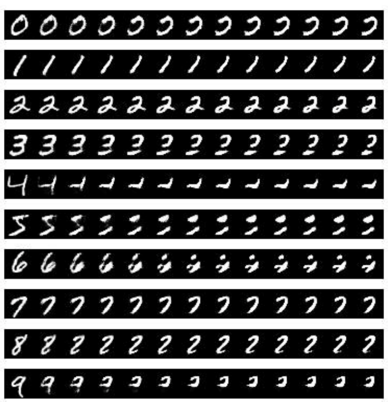


Noise free data on digit 2 model



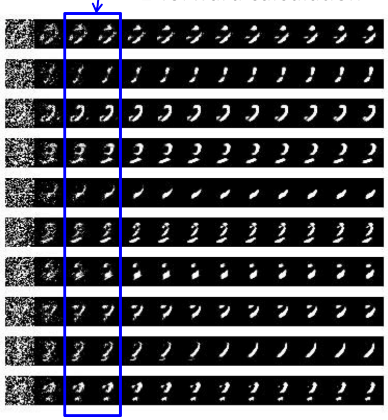
Noisy data (w/ N(0,1/25)) on digit 2 model

RBM as RNN with Very Noisy Input



Noise free data on digit 2 model

may be better than
1-forward calculation



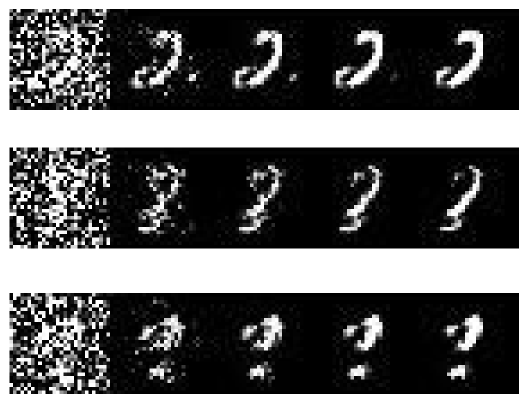
Noisy data (w/ N(0,1)) on digit 2 model

A Closer Look

2

8

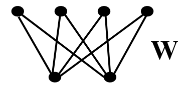
9



recurrent epochs →

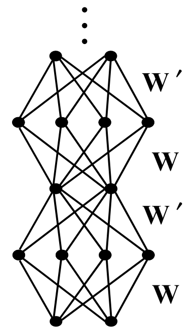
Deep Belief Network & Deep Neural Network

Turning RNN into FNN & Autoencoder



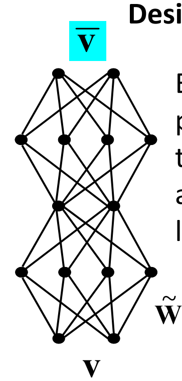
RBM as RNN
Bi-directional connections

Unrolling an RBM



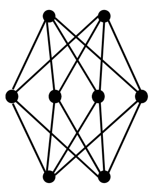
Equivalent uni-directional Feedforward connections

Desired output

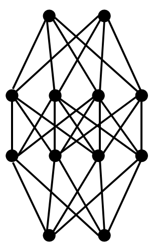


Error propagation to **fine tune** $\tilde{\mathbf{W}}$ at various layers

From Neural Nets to *Deep* Neural Nets



MLP
One hidden layer

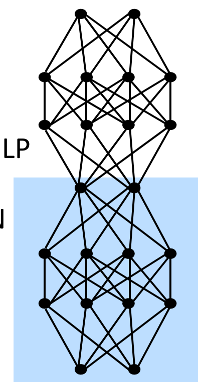


MLP
Many hidden layers

The "vanishing gradient problem," thesis of Hochreiter (1991)

MLP

DBN

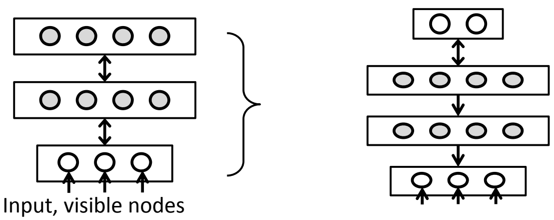


Generative pre-training with RBM

Binary feature, distribution of which can be well approximated by FNN above; data-driven feature extraction

From RBM to Deep Belief Networks

- Deep Belief Networks (DBN): Stacking up layers of RBM and augment final layers with feedforward-like networks
  - Use of RBMs: internal representation of data – **data-driven feature extraction**; noise suppression, kernel transformation → prepare to match data & distribution
  - Use of feedforward nets: logistic regression, function approximation, discrimination, classification, …

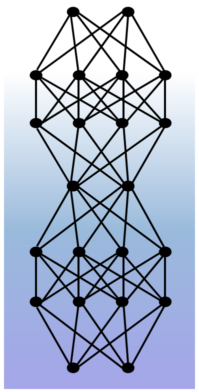


DBN

Deep Neural Nets

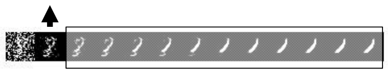
MLP
辨證

DBN
聯想



Generative pre-training with RBM

- Use MLP for explicit decision mapping – take advantage of function approximation capability $\varphi(X) \rightarrow y \in \{C_i\}$
- Middle layers – adjust internal representations (incl. interpolation) to ultimately help minimize decision error
- No longer use RBM as RNN – focus on salient feature selection, suppressing & discarding low correlation components
- Further away from decision layer, less affected by error back propagation, retain saliency in representation



Deep Learning

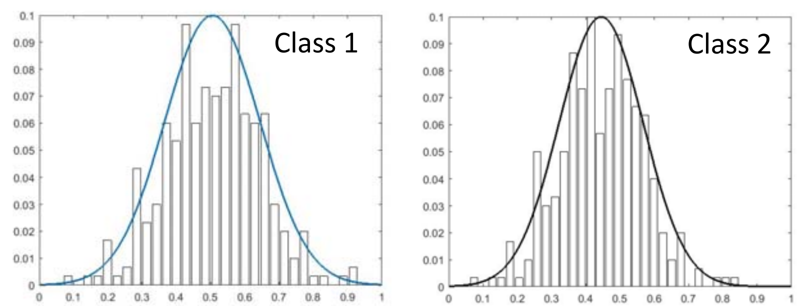
Back to Statistical Pattern Recognition

- Statistical methods are important in data analysis
- Many see statisticians as "data scientists"
- Statistics involves lots of data, but statistics only addresses ONE aspect of data behavior – its distribution, utility of which (e.g., inference) notwithstanding
- Structure of data may exist in a non-trivial manifold

Statistical Pattern Recognition?

C1 = {0.4306, 0.6448, 0.4714, 0.4849, 0.3556, 0.4174, 0.3073, 0.5443, 0.4315, 0.6025, …}

C2 = {0.3789, 0.5677, 0.4149, 0.4267, 0.3129, 0.3673, 0.2705, 0.4793, 0.3799, 0.5303, …}

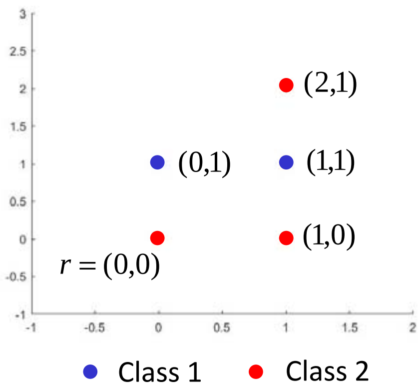


Which Manifold?

C1 = {0.4306, 0.6448, 0.4714, 0.4849, 0.3556, 0.4174, 0.3073, 0.5443, 0.4315, 0.6025, …}
C2 = {0.3789, 0.5677, 0.4149, 0.4267, 0.3129, 0.3673, 0.2705, 0.4793, 0.3799, 0.5303, …}

$$c \rightarrow r = (n_1, n_2)$$
$$n_1 = \mathrm{mod}(10000c, 2)$$
$$n_2 = \mathrm{mod}(10000c, 3)$$

$$\mathrm{C1} = (3n+1)/10000$$
$$\mathrm{C2} = 3n/10000$$
$$\text{or } (2n+1)/10000$$
$$p_E = 1/8$$



Space, Structure, Manifold, What Else?

Given 4-tuples (a, b, c, x), find THE relationship

Empirical data: $(1,2,1,-1)$, $(1,-2,1,1)$, $(4,-4,1,0.5)$, $(1,-8,12,2)$, ...

Machine learning by regression example:

$$x = \sum_{i=0}^{\infty} h_i a^i + \sum_{j=0}^{\infty} g_j b^j + \sum_{k=0}^{\infty} d_k c^k$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \qquad ax^2 + bx + c = 0$$

Final Thoughts

Mixture Models – Worth a Revisit

- Statistical modeling (pdf estimation) followed by discriminative modeling (e.g., training on minimum classification error criterion) has been a common practice for quite some time
- Has Gaussian mixture model been vindicated? Yes, if you understand what RBM/DBN is trying to do
- How to handle large Time-Freq spectral patterns, more than one frame at a time, in statistical models?
  – Work on MRF to alleviate problems in covariance matrix estimation
  – Data reduction techniques that preserve the segmental level of inter-frame correlation
  – Retain representations that can take advantage of many traditional enhancement techniques (signal enhancement, normalization, adaptation in feature space, …)