On Unsupervised Feature Learning with Deep Neural Networks

Huan Sun

Dept. of Computer Science, UCSB

Warm Thanks To

- Committee
  - **Prof. Xifeng Yan**
  - **Prof. Linda Petzold**
  - **Prof. Ambuj Singh**

Outline

- Introduction
- A New Generation of Neural Networks
- Neural Networks & Biclustering
- Preliminary Results
- Future Work

Outline

- **Introduction**
- A New Generation of Neural Networks
- Neural Networks & Biclustering
- Preliminary Results
- Future Work

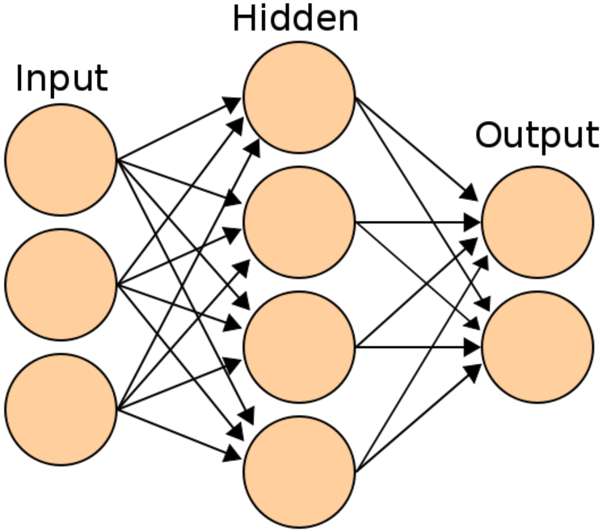
Neural Networks

- What are neural networks?

- What can we do with neural networks?

Neural Networks

- What are neural networks?
  - **Computational model**
  - **Inspired by biological neural networks**

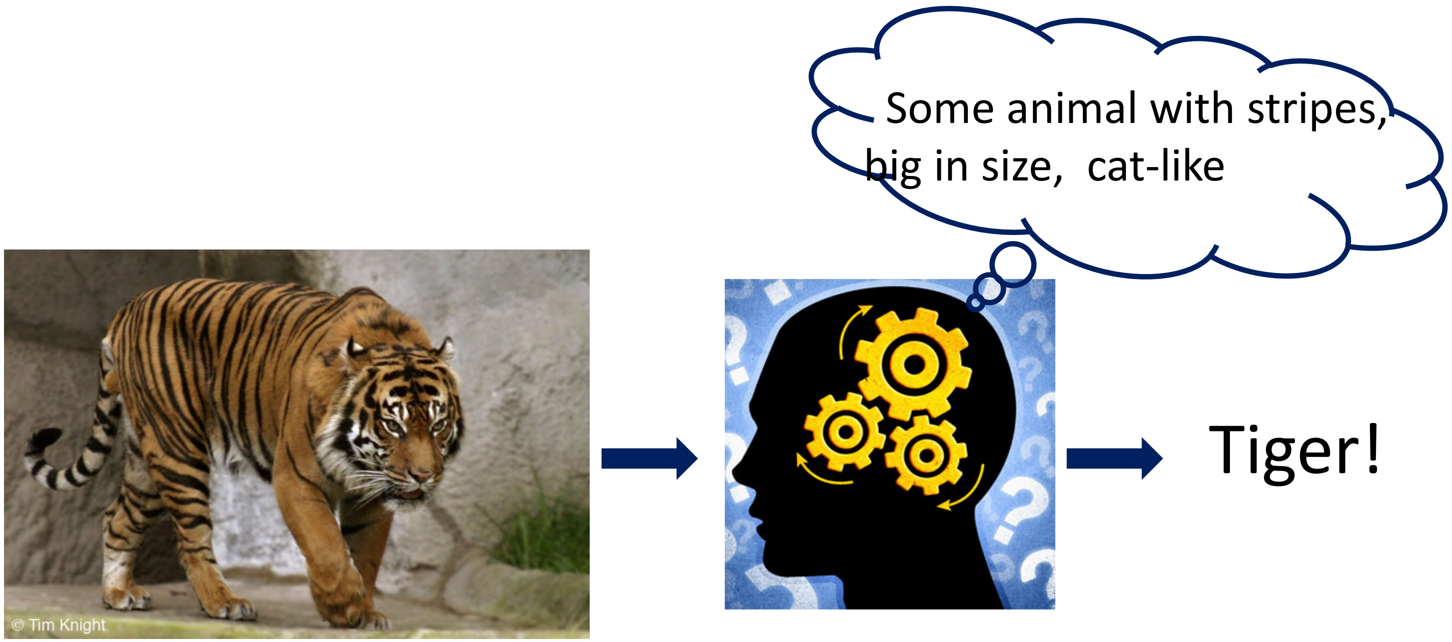


Neural networks in a brain

- What can we do with neural networks?
  - **Regression analysis**
  - **Classification (including pattern recognition)**
  - **Data processing (e.g. clustering)**

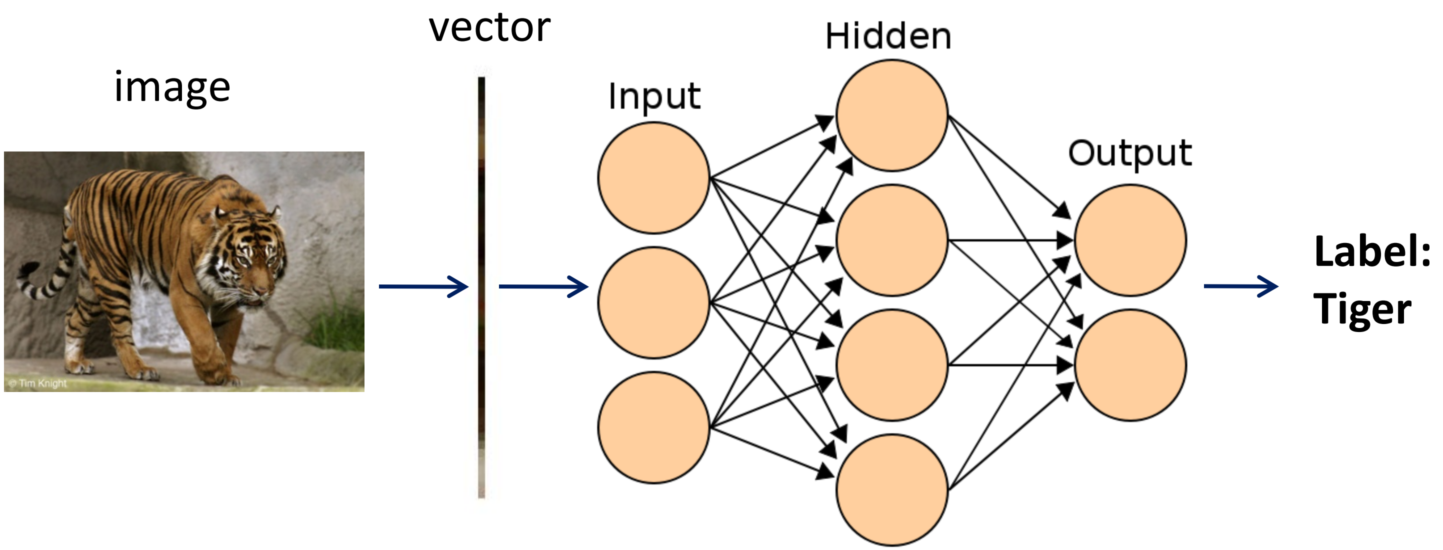
Aim of Neural Networks

- Humans better at recognizing patterns than computers



Aim of Neural Networks

- Humans better at recognizing patterns than computers
- Can we train computers by mimicking the brain?

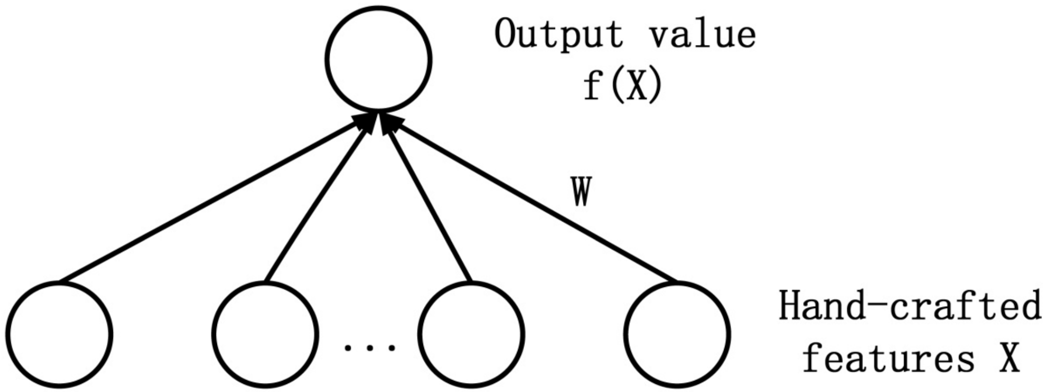


Artificial neural networks

History of Neural Networks

- First Generation (1960s)
  - Perceptron

Illustration:



Input: {(x, t),…}, where $\mathrm{x} \in \Re^n$, t $\in$ {+1, −1}

Output:

classification function f(x)=w’*x+b

such that f(x)>0 => t=1 and f(x)<0 => t=-1

History of Neural Networks

- First Generation (1960s)
  - Perceptron

Algorithm:

- Initialize: w, b
- For each sample x (data point)

  Predict the label of instance x to be y = sign(f(x))

  If y$\neq$t, update the parameters by gradient descent

  $$w \leftarrow w - \eta\,(\nabla_w E) \quad \text{and} \quad b \leftarrow b - \eta\,(\nabla_b E)$$

  Else w and b does not change
- Repeat until convergence

Note: E is the cost function to penalize the mistakes,

e.g. $$E = \sum_k \left(t_k - f(x_k)\right)^2$$

History of Neural Networks

- First Generation (1960s)
  - Perceptron

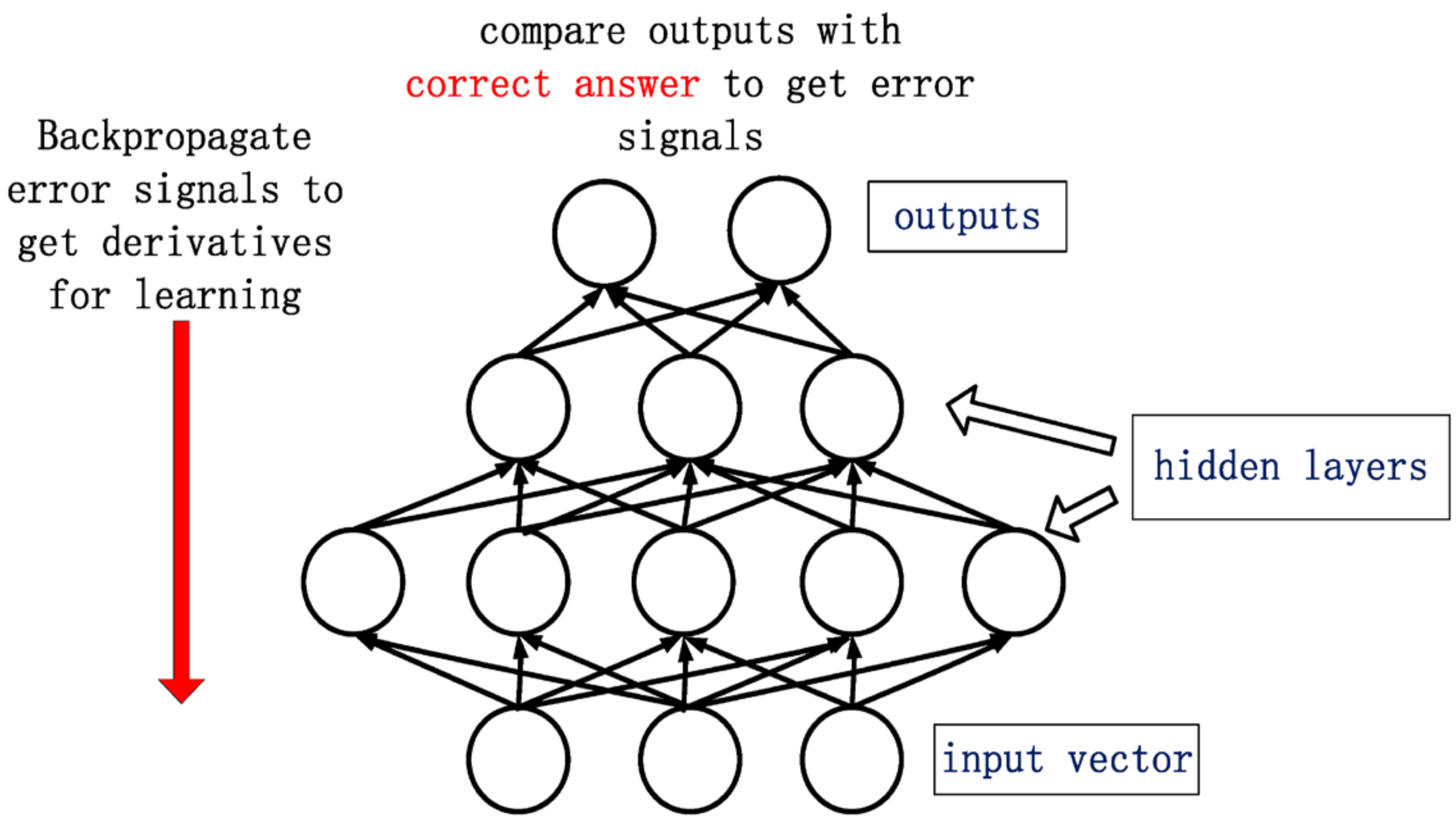
Example: Object (e.g. tiger) classification

- $\mathrm{x} = (x_1, x_2, x_3, \ldots, x_n)$, $t = +1$
  - x_1 : existence of strips
  - x_2 : similarity to a cat
  - ...
- Output $f(\mathrm{x})$ *such that* $f(\mathrm{x})>0$ => tiger and $f(\mathrm{x})<0$ => not tiger

The input features are pre-obtained hand-crafted features from the original data, and not adaptable during training the model.

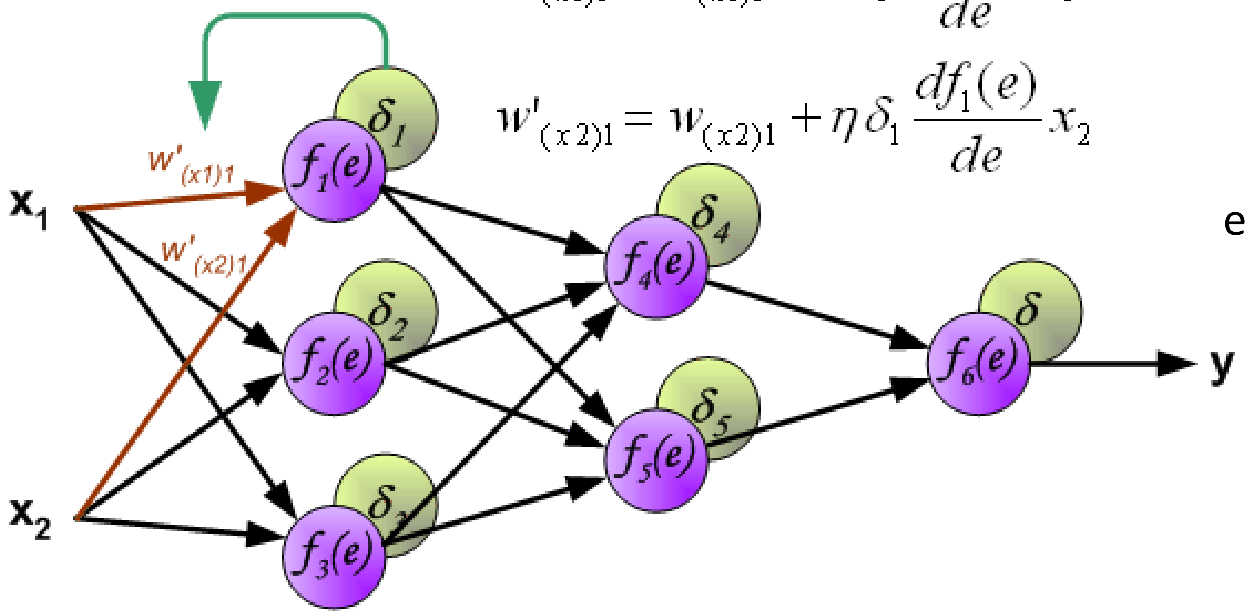
History of Neural Networks

- First Generation (1960s)
  - Perceptron
- Second Generation (1980s)
  - Backpropagation



Problems with Backpropagation

- Require a large amount of labeled data in training
- Backpropagation in a deep network (with >=2 hidden layers)



$$w'_{(x1)1} = w_{(x1)1} + \eta \delta_1 \frac{df_1(e)}{de} x_1$$

$$w'_{(x2)1} = w_{(x2)1} + \eta \delta_1 \frac{df_1(e)}{de} x_2$$

e.g.

$$\delta = y - t$$

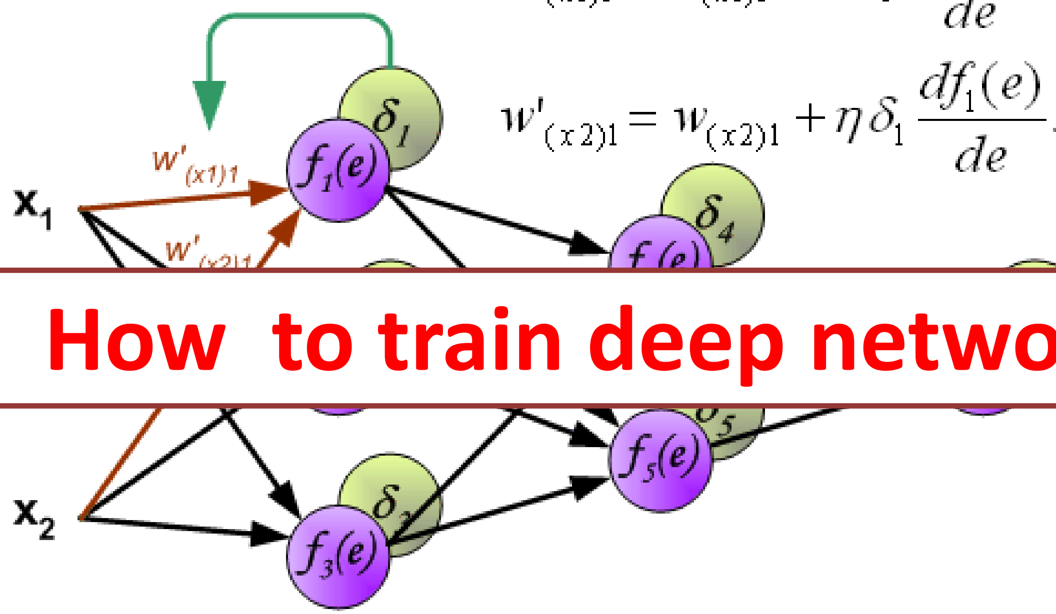
Backpropagated errors (δ’s) to the first few layers will be minuscule , therefore updating tend to be ineffectual.

Problems with Backpropagation

- Require a large amount of labeled data in training
- Backpropagation in a deep network (with >=2 hidden layers)

$$w'_{(x1)1} = w_{(x1)1} + \eta \delta_1 \frac{df_1(e)}{de} x_1$$

$$w'_{(x2)1} = w_{(x2)1} + \eta \delta_1 \frac{df_1(e)}{de} x_2$$



How to train deep networks?

Backpropagated errors (δ's) to the first few layers will be minuscule , therefore updating tend to be ineffectual.

Stuck in training …

- Limited power of a shallow neural network
- Less insights about the benefits of more layers
- Popularity of other tools, such as SVM

=> Less research works on neural networks

Breakthrough

- Reducing the Dimensionality of Data with Neural Networks (Hinton *et al*., Science, 2006)
  - successfully train a neural network with 3 or more hidden layers
  - more effective than Principal Component Analysis (PCA) etc.

- A new generation: emergence of research works on deep neural networks

Outline

- **Introduction**
- A New Generation of Neural Networks
- Neural Networks & Biclustering
- Preliminary Results
- Future Work

Related Work of Deep Neural Networks

- Training algorithms

- Applications

Related Work of Deep Neural Networks

- Training algorithms
  - Reducing the Dimensionality of Data with Neural Networks (Hinton *et al.*, Science, 2006)
  - Others
- Applications
  - Text
  - Vision
  - Audio

Related Work of Deep Neural Networks

- **Training algorithms**
  - **Reducing the Dimensionality of Data with Neural Networks (Hinton et al., Science, 2006)**
  - **Others**
- **Applications**
  - **Text**
  - **Vision**
  - **Audio**

Text (1): sentiment distribution prediction (Socher *et al.*, EMNLP'11)

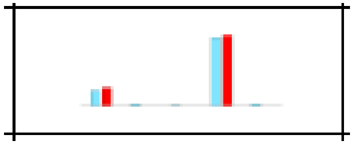
- **Problem description**

Given a personal story, predict its sentiment distribution.

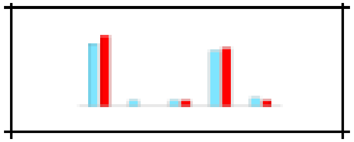
e.g. 5 sentiment classes are [*Sorry, Hugs; You Rock (approvement); Teehee (amusement); I Understand; Wow, Just Wow (shock)*]

Stories | **Predicted (light blue) & true (red)**

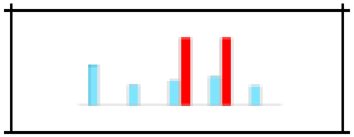
1. **I wish I knew someone to talk to here.**



2. **I loved her but I screwed it up. Now she's moved on. I will never have her again. I don't know if I will ever stop thinking about her.**



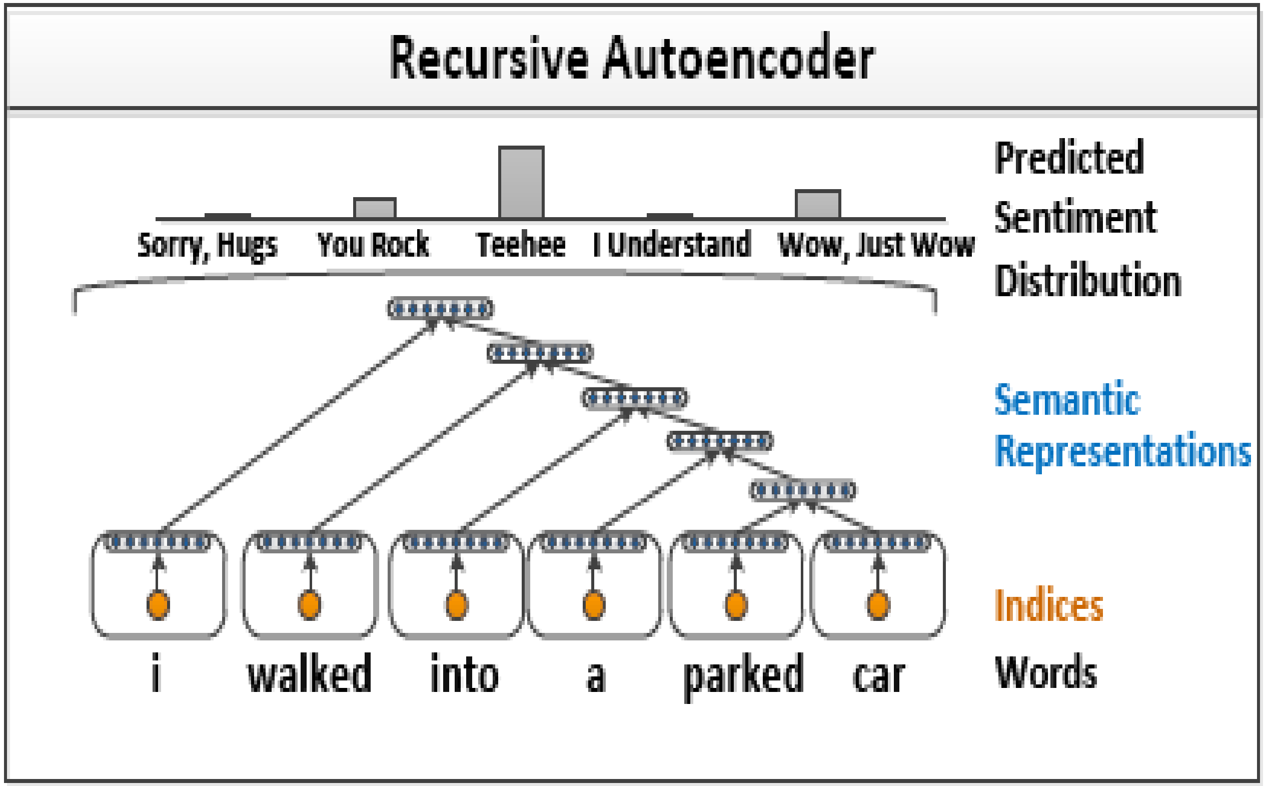
3. **My paper is due in less than 24 hours and I'm still dancing around the room.**

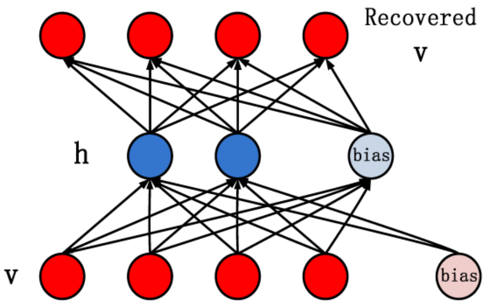


Text (1): sentiment distribution prediction (Socher *et al.*, EMNLP'11)

- **Model Illustration**

A deep neural network: Recursive Autoencoder



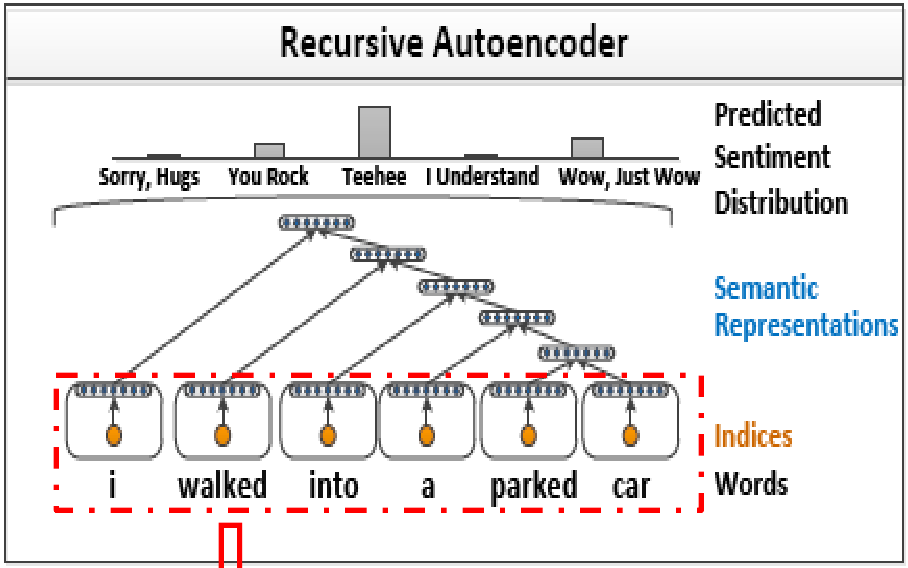


autoencoder

Text (1): sentiment distribution prediction (Socher *et al.*, EMNLP'11)

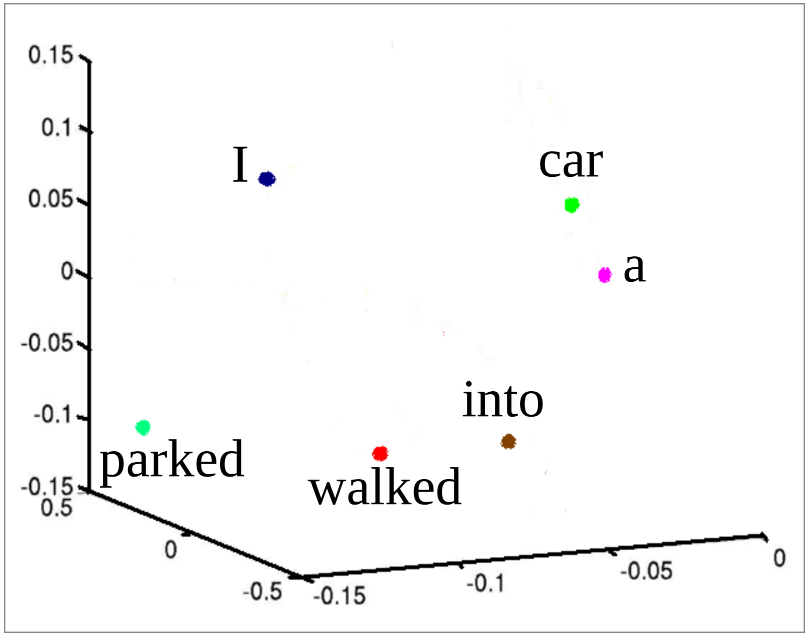
- **Model Illustration**

A deep neural network: Recursive Autoencoder



Map each word to $\Re^n$, e.g. n=3, by

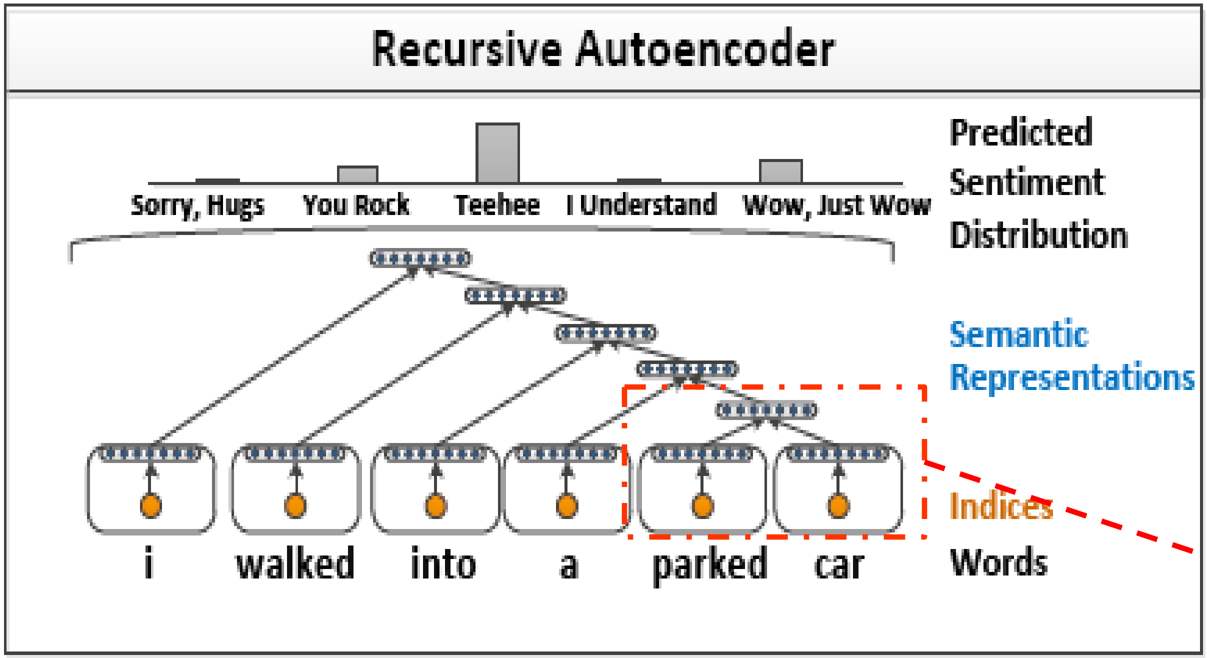
Random initialization; Or pre-processing with existing language models

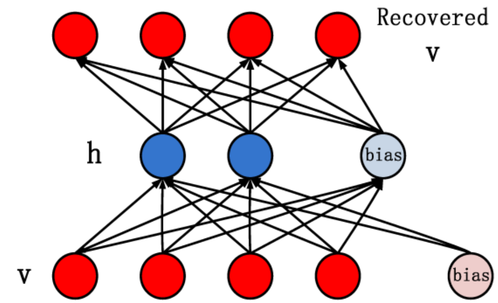


Text (1): sentiment distribution prediction (Socher *et al.*, EMNLP'11)

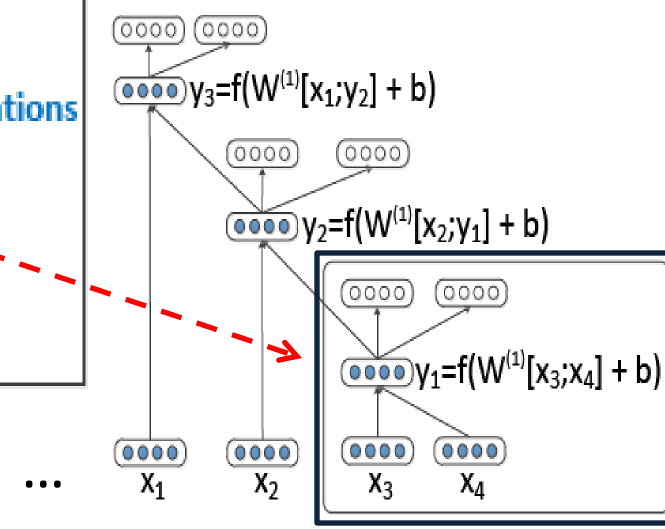
- **Model Illustration**

A deep neural network: Recursive Autoencoder





autoencoder



Q: Which two words to combine?

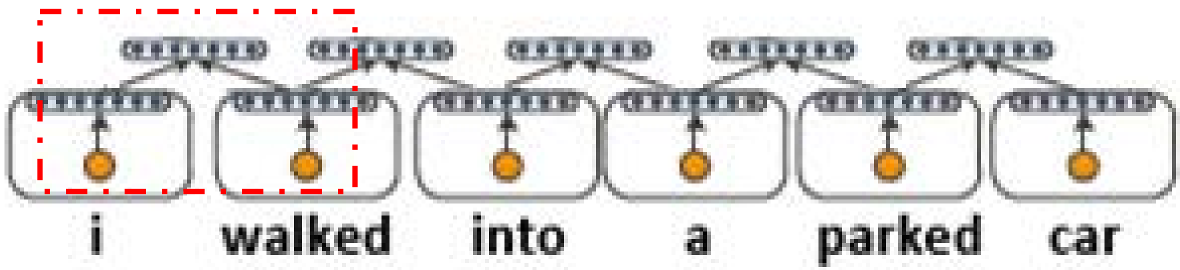
Text (1): sentiment distribution prediction (Socher *et al.*, EMNLP'11)

- **Model Illustration**

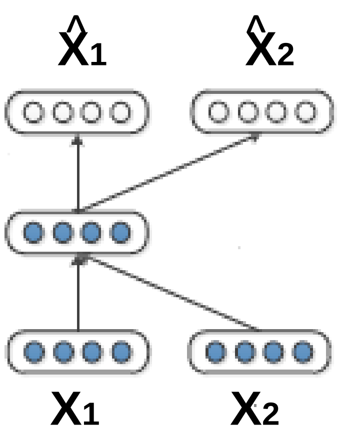
A deep neural network: Recursive Autoencoder

Q: Which two words to combine?

Combine every two neighboring words with an autoencoder,



e. g.



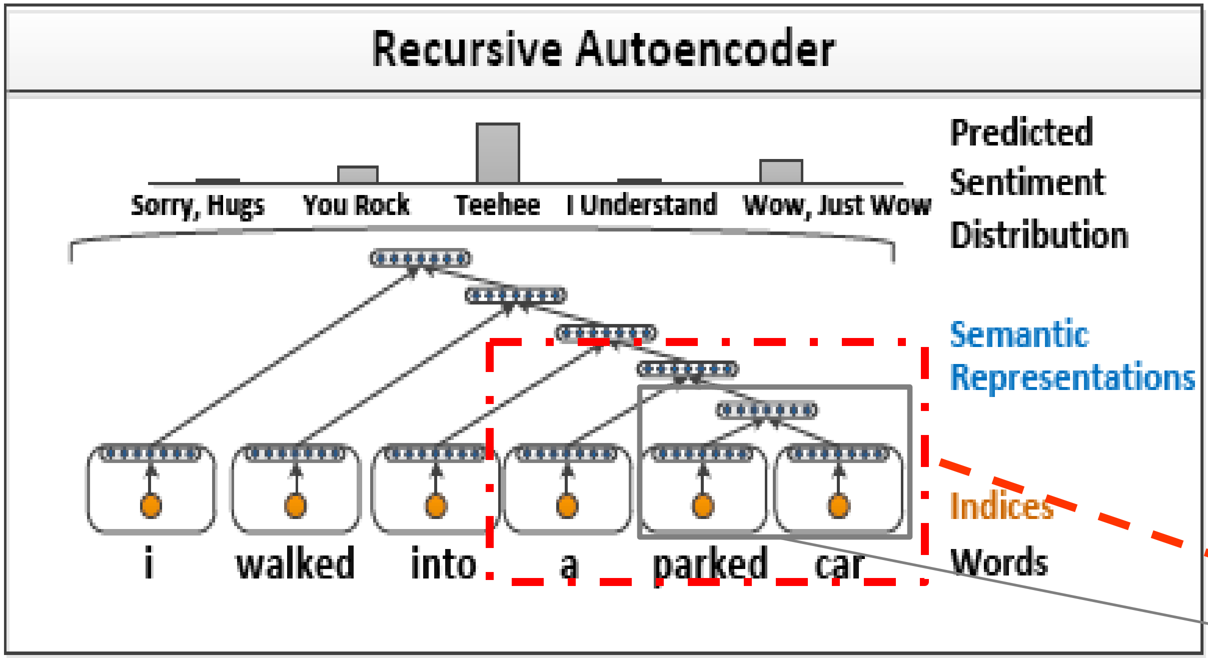
Reconstruction error: $\left\| [\hat{X}_1; \hat{X}_2] - [X_1; X_2] \right\|_2^2$

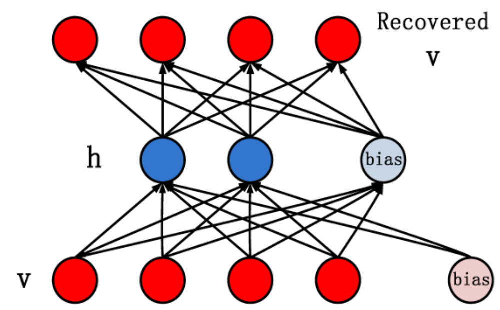
Select the word pair with the lowest reconstruction error, here it is "parked car".

Text (1): sentiment distribution prediction (Socher *et al.*, EMNLP'11)

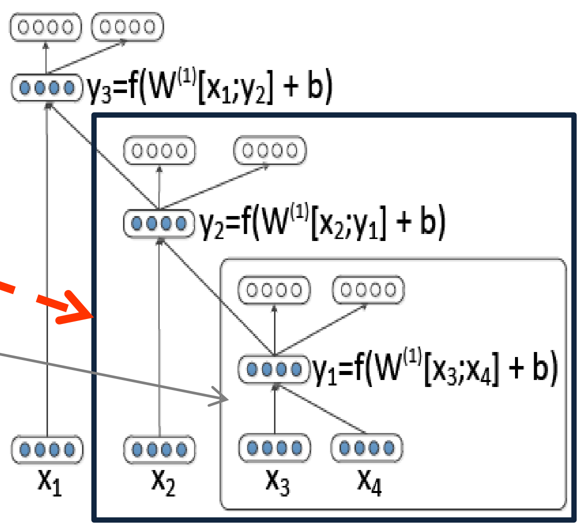
- **Model Illustration**

A deep neural network: Recursive Autoencoder





autoencoder

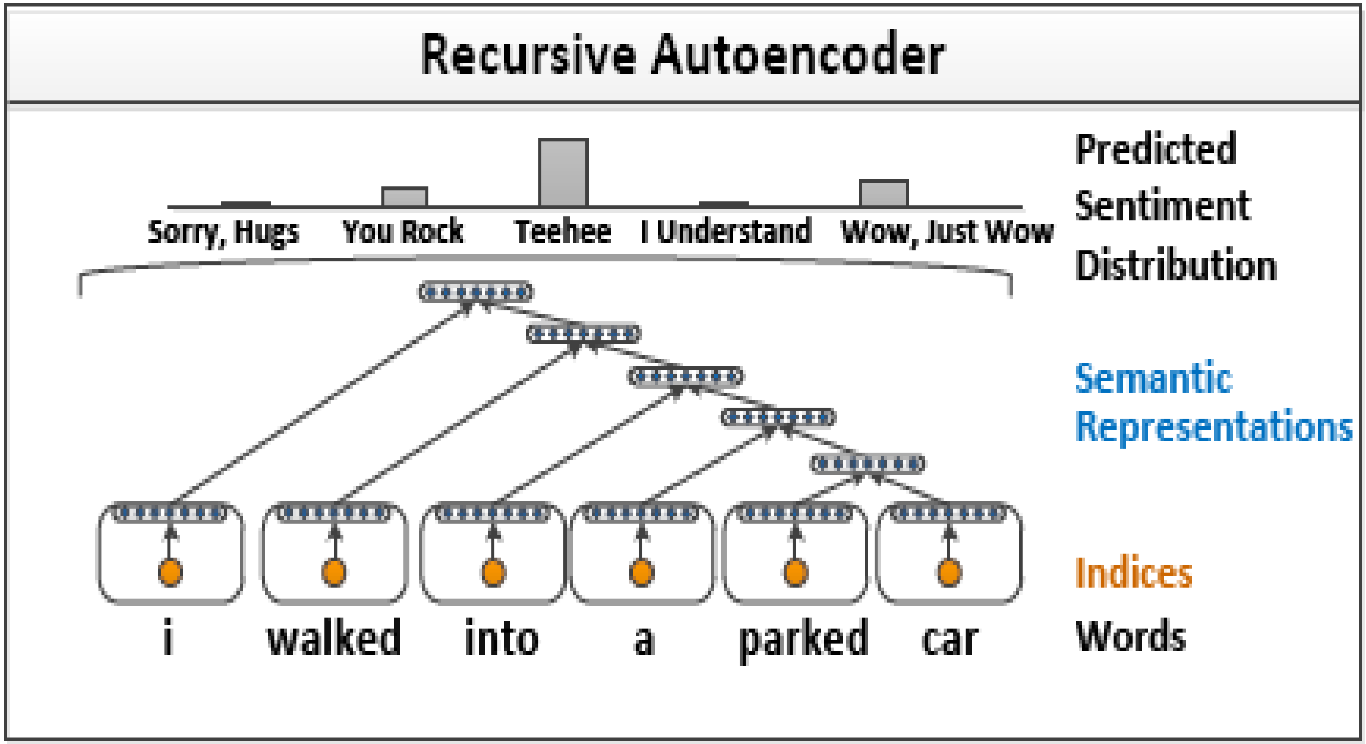


- **The parent node for "parked car" is regarded as a new word.**
- **Recursively learn a higher-level representation using an autoencoder**

Text (1): sentiment distribution prediction (Socher *et al.*, EMNLP'11)

- **Model Illustration**

A deep neural network: Recursive Autoencoder



- **Instead of using a bag-of-words model, exploit hierarchical structure and use compositional semantics to understand sentiment**

Text (2): paraphrase detection (Socher *et al*., NIPS'11)

- **Problem description**

Given two sentences, predict whether they are paraphrase of each other

e.g.

1. The judge also refused to postpone the trial date of Sept. 29.

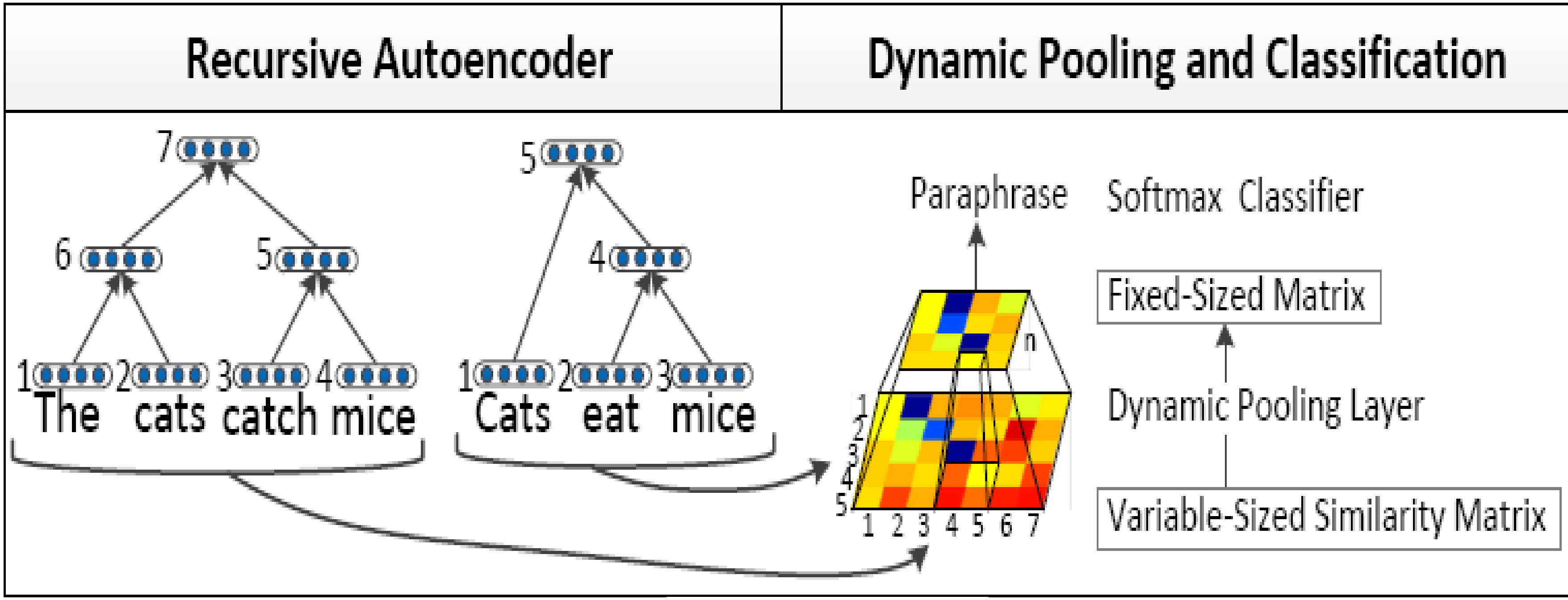
2. Obus also denied a defense motion to postpone the September trial date.

Model	Acc.	F1
All Paraphrase Baseline	66.5	79.9
Rus et al. (2008) [16]	70.6	80.5
Mihalcea et al. (2006) [17]	70.3	81.3
Islam and Inkpen (2007) [18]	72.6	81.3
Qiu et al. (2006) [19]	72.0	81.6
Fernando and Stevenson (2008) [20]	74.1	82.4
Wan et al. (2006) [21]	75.6	83.0
Das and Smith (2009) [15]	73.9	82.3
Das and Smith (2009) + 18 Features	76.1	82.7
$\boxed{\text{Unfolding RAE + Dynamic Pooling}}$	**76.8**	**83.6**

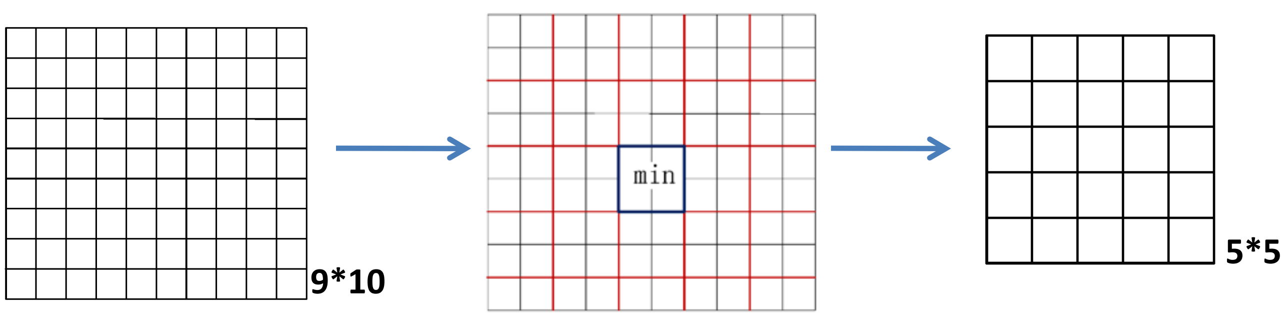
Text (2): paraphrase detection(Socher *et al.*, NIPS'11)

- **Model Illustration**

  Recursive autoencoder with dynamic pooling



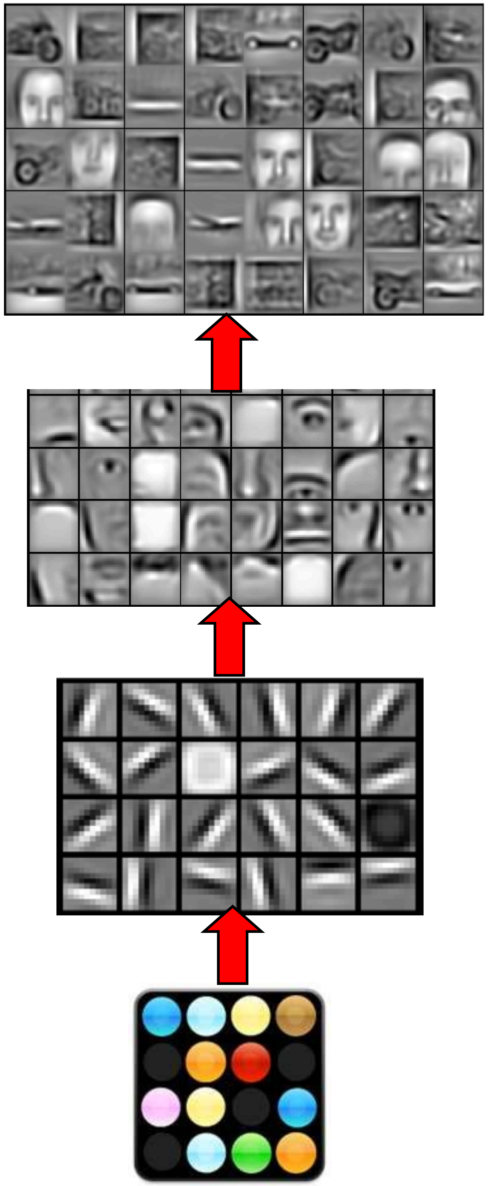
e.g.
pooling



Vision: convolutional deep belief networks (Lee *et al.*, NIPS'09)

- **Problem description**
  - ◆ To learn a hierarchical model that represents multiple levels of visual world
  - ◆ Scalable to realistic images (~200*200)

- **Advantages**
  - ◆ Appropriate for classification, recognition
  - ◆ Both specific and general-purpose than hand-crafted features



Vision: convolutional deep belief networks (Lee *et al.*, NIPS'09)

- **Model structure**
  - **Each layer configuration:**

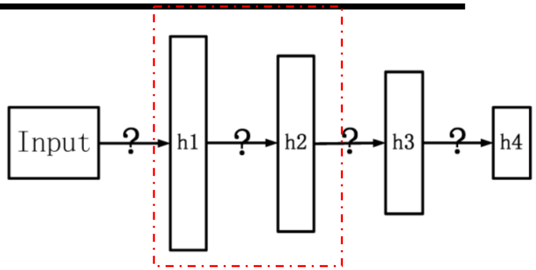
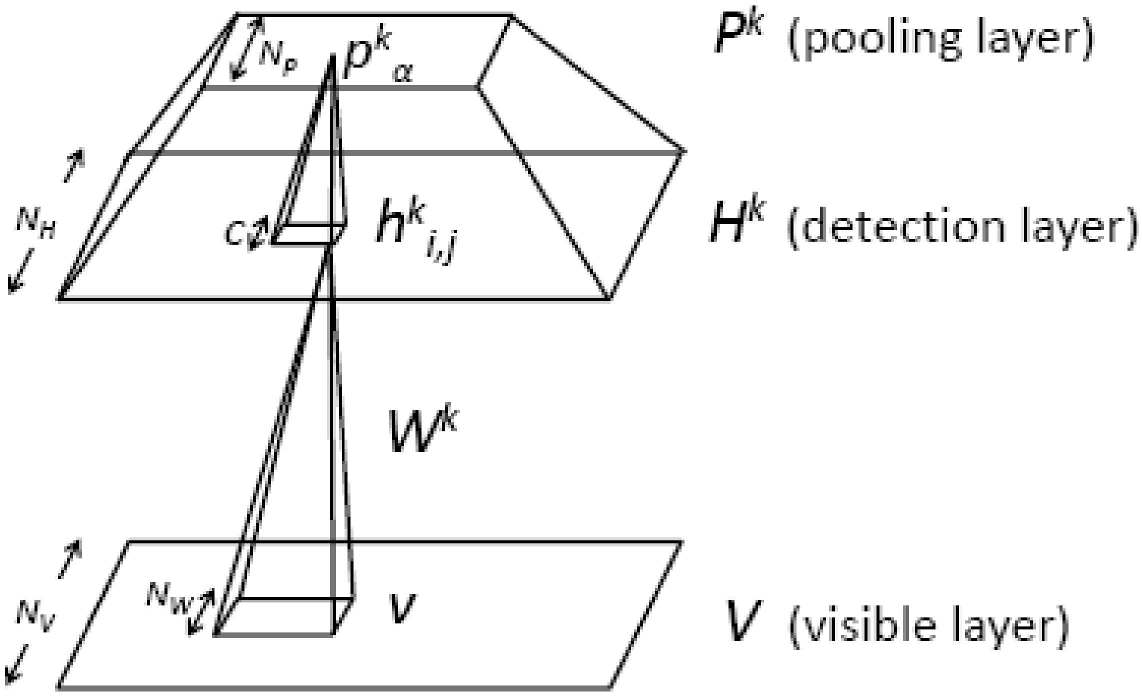


Fig. 1 General look

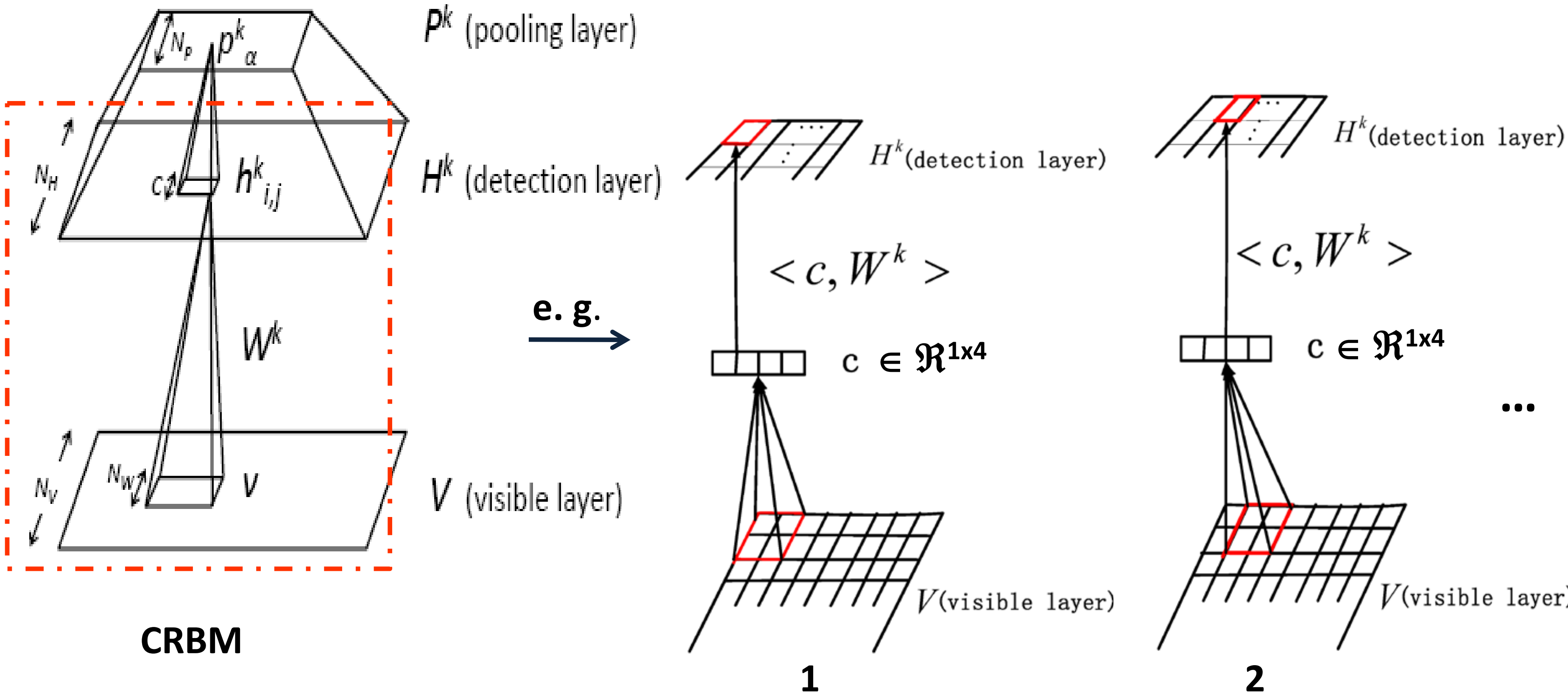


Convolutional Restricted Boltzman Machine (CRBM)

  - **Stack CRBM one by one to form the deep networks**

Vision: convolutional deep belief networks (Lee *et al.*, NIPS'09)

- **Model structure**
  - **Each layer configuration:**



  - **Stack CRBM one by one to form the deep networks**

Related Work of Deep Neural Networks

- **Training algorithms**
  - **Reducing the Dimensionality of Data with Neural Networks (Hinton et al., Science, 2006)**
  - **Others**
- Applications
  - Text
  - Vision
  - Audio

Three Ideas in [Hinton *et al.*, Science, 2006]

- To learn a model that generates the input data rather than classifying it: no need for a large amount of labeled data;

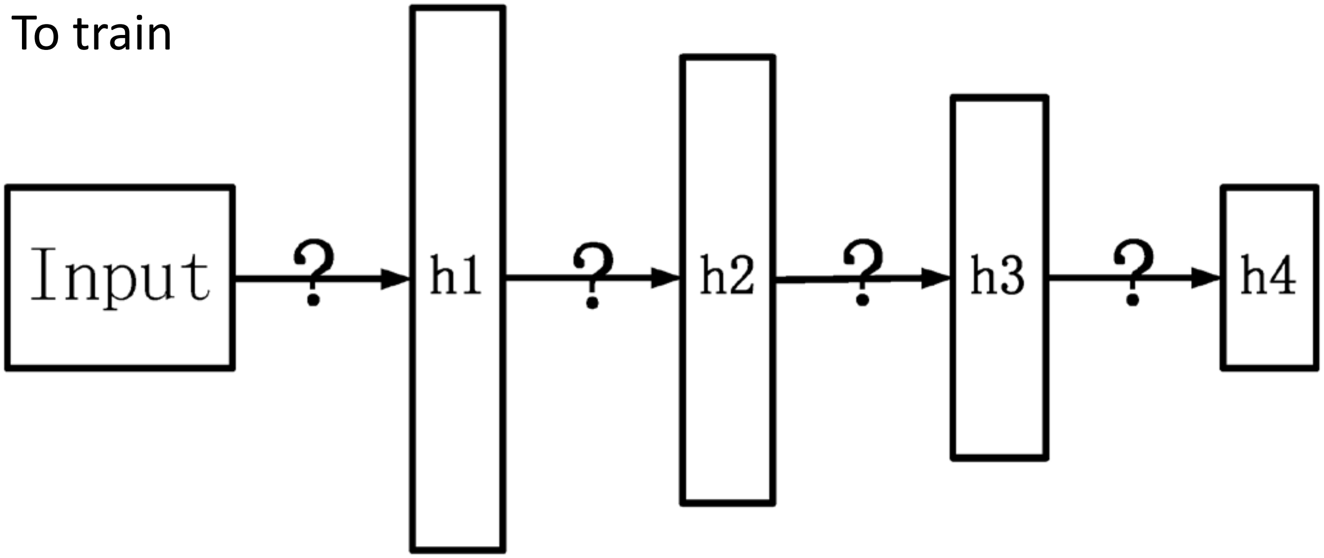
- To learn one layer of representation at a time: decompose the overall learning task to multiple simpler tasks;

- To use a separate fine-tuning stage : further improve the generative/discriminative abilities of the composite model.

Training Deep Neural Networks

- **Procedure** (Hinton et al., Science, 2006)
  - Unsupervised layer-wise pre-training
  - Fine-tuning with backpropagation
- **Example**

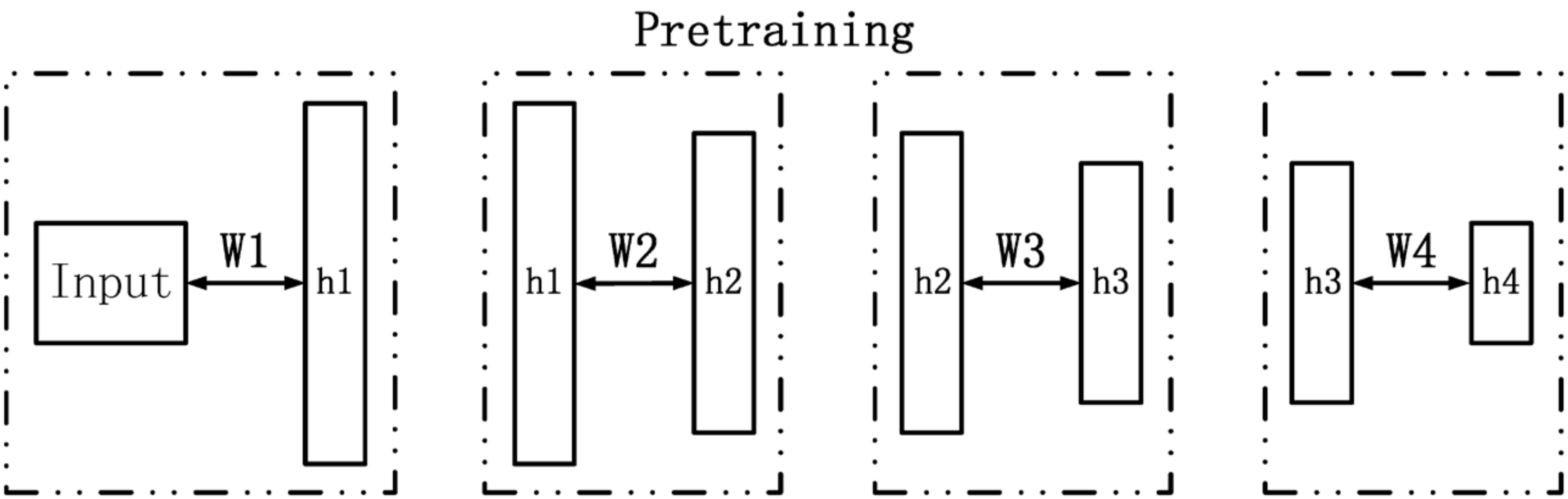
To train



Training Deep Neural Networks

- **Procedure**(Hinton et al., Science, 2006)
  - ◆ **<u>Unsupervised layer-wise pre-training</u>**
    - ✓ Restricted Boltzmann Machine (RBM)
  - ◆ Fine-tuning with backpropagation

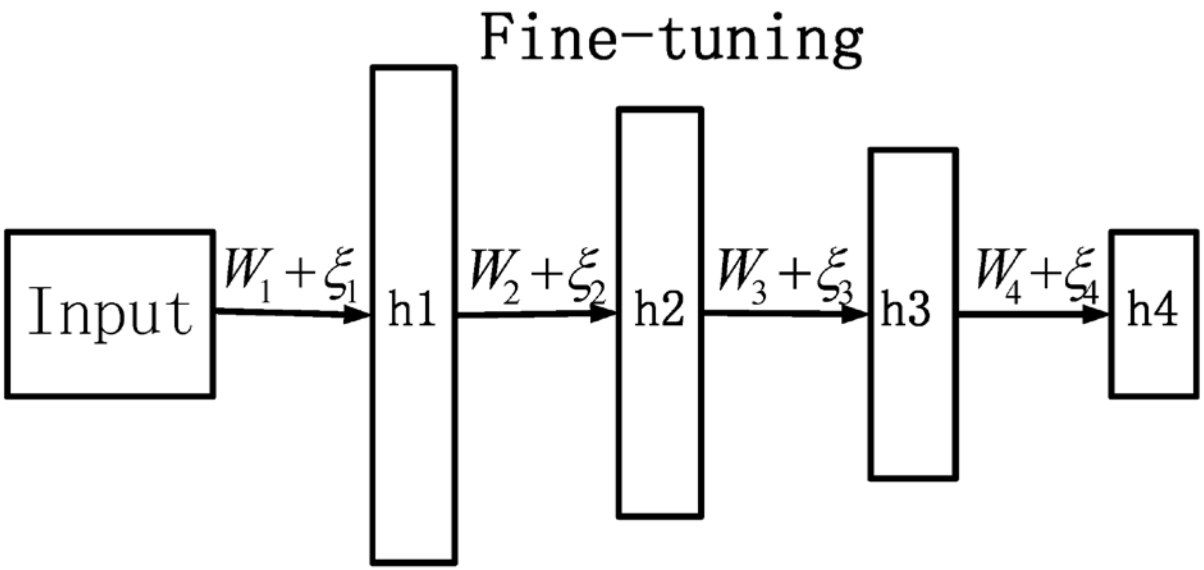
- **Example**



Training Deep Neural Networks

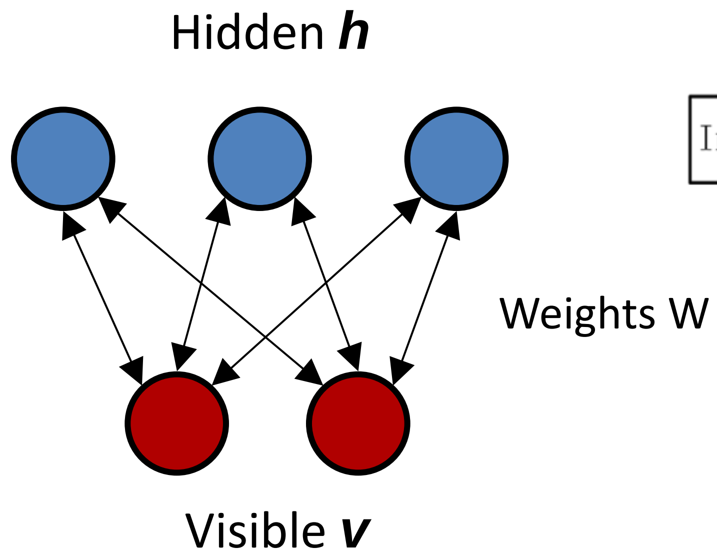
- **Procedure** (Hinton et al., Science, 2006)
  - ◆ Unsupervised layer-wise pre-training
    - ✓ Restricted Boltzmann Machine (RBM)
  - ◆ **Fine-tuning with backpropagation**

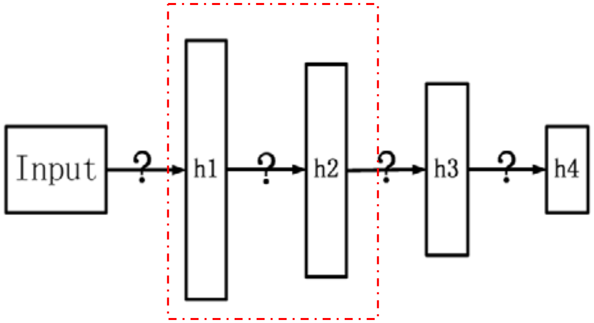
- **Example**



Layer-Wise Pre-training

- **A learning module: restricted Boltzman machine (RBM)**

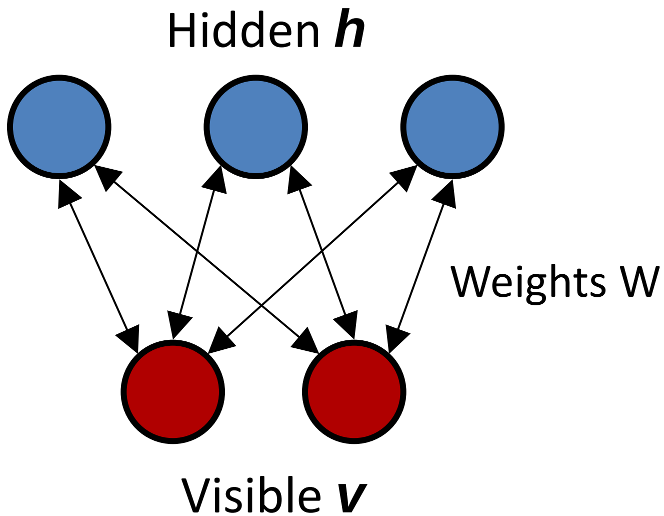




- **only one layer of hidden units**
- **no connections inside each layer**
- **the hidden (visible) units are independent given the visible (hidden) units**

Layer-Wise Pre-training

- **A learning module: restricted Boltzman machine (RBM)**



$$E(\mathbf{v},\mathbf{h}) = -\sum_{i\in \text{pixels}} b_i v_i - \sum_{j\in \text{features}} b_j h_j - \sum_{i,j} v_i h_j w_{ij}$$

- **Weights -> Energies -> Probabilities**
  - **Each possible joint configuration of the visible and hidden units has an "energy" : determined by weights and biases**
  - **The energy determines the probability of choosing such configuration**

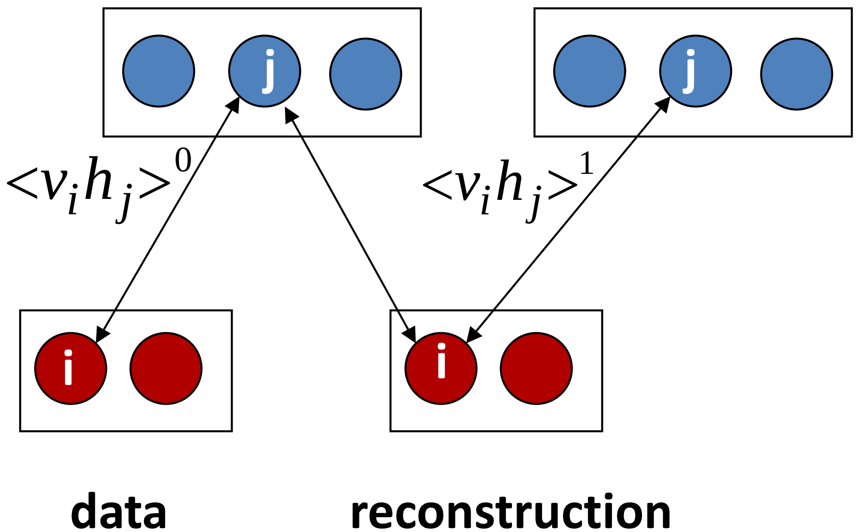
$$P(\mathbf{v},\mathbf{h}) = \frac{1}{Z}\exp(-E(\mathbf{v},\mathbf{h}))$$

- **Objective function:**

$$\max \mathrm{P}(v) = \max \sum_h \mathrm{P}(v,h)$$

Layer-Wise Pre-training

- **Alternate Gibbs sampling to learn the weights of an RBM**



1. Start with a training vector on the visible units.

2. Update all the hidden units in parallel

3. Update all the visible units in parallel to get a "reconstruction".

4. Update all the hidden units again.

Contrastive Divergence

$$\Delta w_{ij} = \varepsilon \left(<v_i h_j>^0 - <v_i h_j>^1 \right)$$

- **where < > means the frequency with which neuron i and neuron j are on (with value 1) together;**
- **approximation to the true gradient of the likelihood** $\mathrm{P}(v)$

Training a Deep Neural network

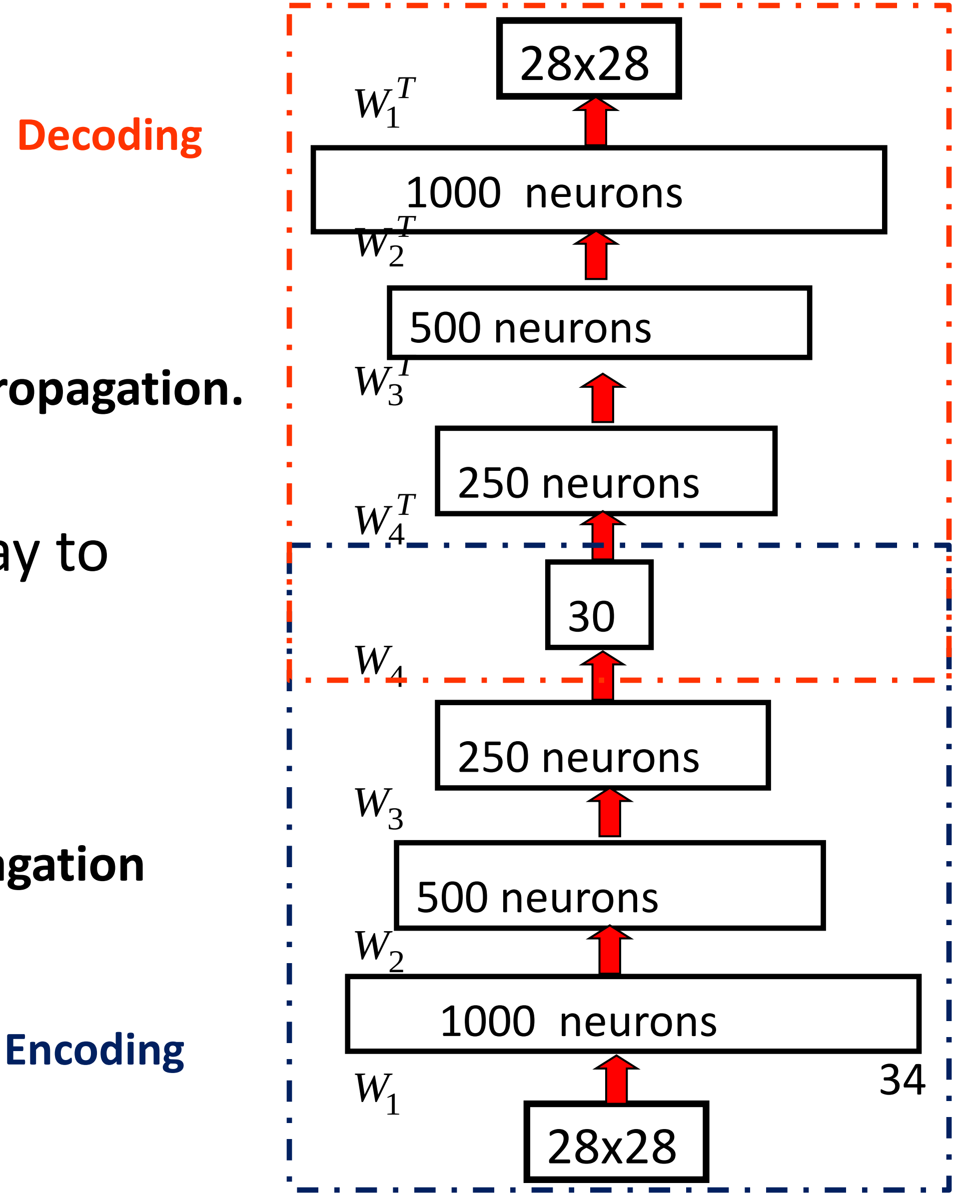
- First train a layer of features that receive input directly from the original data (pixels).

- Then use the output of the previous layer as the input for the current layer, and train the current layer as an RBM

- Fine-tune with backpropagation
  - **Do not start backpropagation until we have sensible weights that already do well at the task**
  - **The label information (if any) is only used in the final fine-tuning stage (to slightly modify the features)**

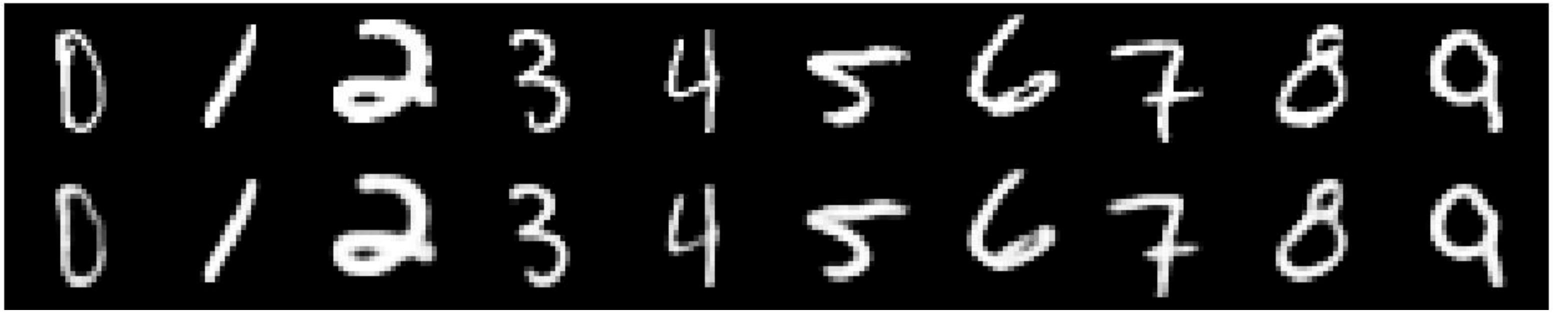
Example: Deep Autoencoders

- A nice way to do non-linear dimensionality reduction:
  - ***very* difficult to optimize deep autoencoders directly using backpropagation.**
- We now have a much better way to optimize them:
  - **First train a stack of 4 RBM's**
  - **Then "unroll" them.**
  - **Finally fine-tune with backpropagation**



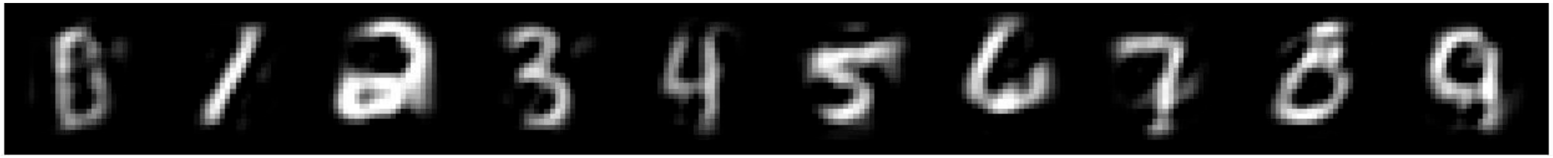
Example: Deep Autoencoders

- A comparison of methods for compressing digit images to 30 dimensions.



real data

30-D deep autoencoder



30-D PCA

Significance

- **Layer-wise pre-training initializes parameters in a good local optimum.** (Erhan et al., JMLR'10)

- **Training deep neural networks both effectively and fast**

- **Unsupervised learning: no need to have labels**

- **Hierarchical structure: more similar to learning in brains**

What can we do?

- Apply neural networks outside text/vision/audio

- Learn semantic features in text analysis to replace traditional language models

- Automatic text annotation for image segments

- Multiple object (unknown sizes) recognition in images

- Model robustness against noise (such as incorrect grammars, not complete sentences, occlusion in images)

- ...

Our Work

- Apply neural networks outside text/vision/audio
  - **gene expression (microarray) analysis**

- Learn semantic features in text analysis to replace traditional language models

- Automatic text annotation for image segments

- Multiple object (unknown sizes) recognition in images

- Model robustness against noise (such as incorrect grammars, not complete sentences, occlusion in images)

- …

Application to Microarray Analysis

Neural Networks:
Feature learning
Autoencoder
Recursive autoencoder
Convolutional autoencoder
....
....

↔

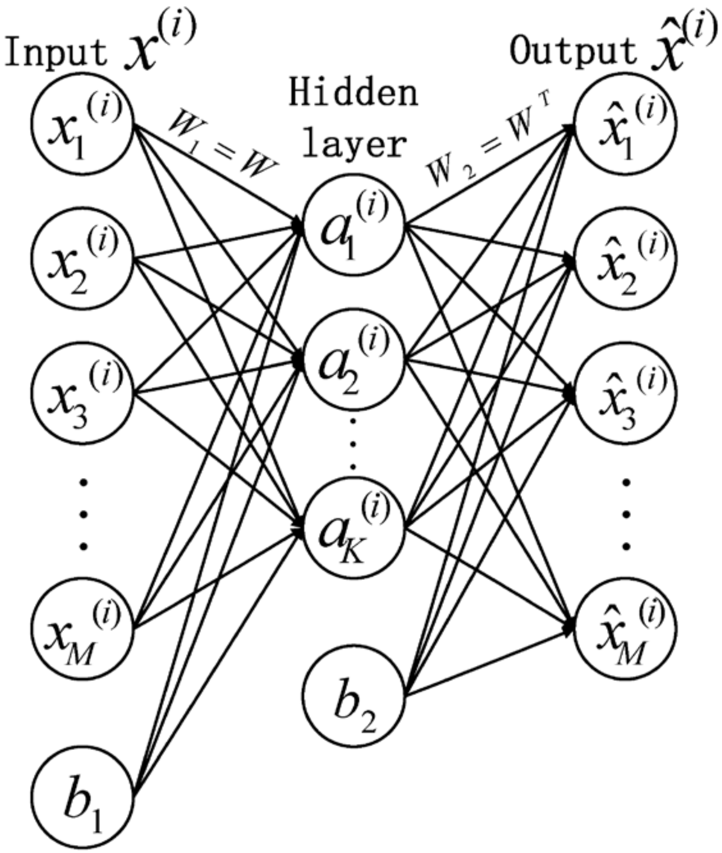
Microarray analysis:
Biclustering
Combinatorial algorithms
Generative approaches
Matrix factorization
....
....

Outline

- **Introduction**
- A New Generation of Neural Networks
- Nerual Networks & Biclustering
- Preliminary Results
- Future Work

Autoencoder (Hinton *et al.*, Science, 2006)

- **Two-layer neural network**



Input: $X = [x^{(1)}, \cdots, x^{(i)}, \cdots, x^{(N)}]$

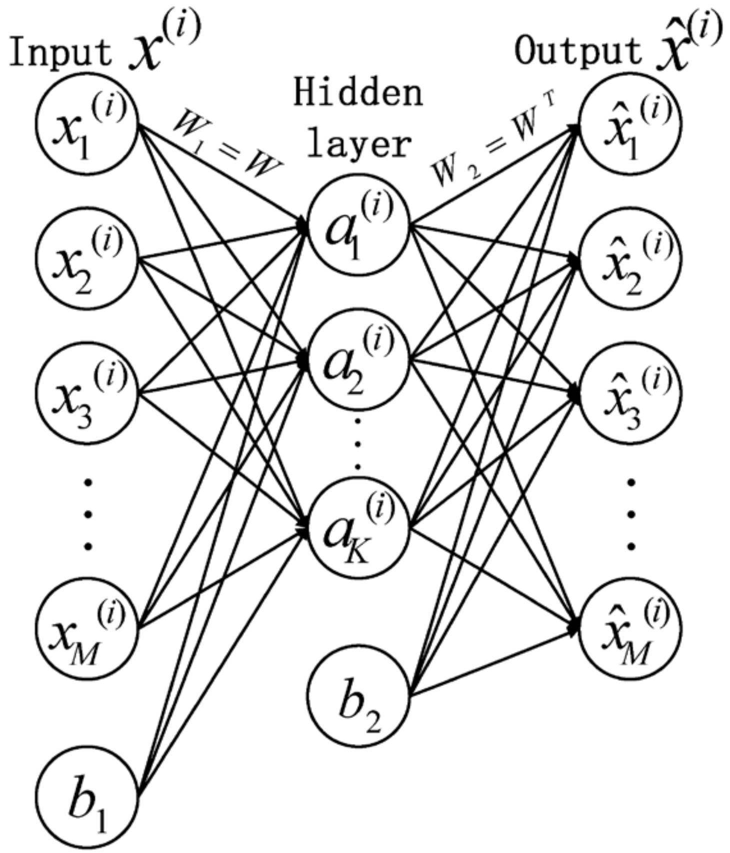
Output:
recovered data $\hat{X}$
weights W
activation value $A = [a^{(1)}, \cdots, a^{(i)}, \cdots, a^{(N)}]$

Optimization formulation:

$$\underset{W, b_1, b_2}{argmin} \quad H = \frac{1}{2N} * \sum_{n=1}^{N} \sum_{m=1}^{M} (\hat{x}_m^{(n)} - x_m^{(n)})^2 \quad (i)$$
$$+ \frac{\lambda}{2} * \|W\|_F^2 \quad (ii)$$

Sparse Autoencoder (Lee *et al.*, NIPS'08)

- **Two-layer neural network**



$a^{(i)}$: K*1 vector of a sigmoid output ,
i.e. $a^{(i)} = sigmoid(W * x^{(i)} + b_1)$

Define the activation rate of hidden neuron k:

$$\hat{\rho}_k = \sum_{i=1}^{N} a_k^{(i)} / N$$

Optimization formulation:

$$\underset{W,b_1,b_2}{argmin} \quad H = \frac{1}{2N} * \sum_{n=1}^{N} \sum_{m=1}^{M} (\hat{x}_m^{(n)} - x_m^{(n)})^2 \quad (i)$$
$$+ \beta_2 * KL(\rho \| \hat{\rho}) \quad (ii)$$
$$+ \frac{\lambda}{2} * \|W\|_F^2 \quad (iii)$$

Biclustering Review

- Simultaneously group genes and conditions in a microarray

(Cheng and Church, ISMB'00)

Genes (columns) × Conditions (rows)

0	0	0	0	0	0	0	0
0	1	1	1	0	-1	0	0
0	-1	0	-1	0	0	0	1
0	1	1	1	-1	-1	1	0
0	-1	-1	0	1	1	-1	0
0	0	1	1	-1	0	1	0
0	0	-1	-1	1	1	-1	0
1	0	0	0	0	0	0	0

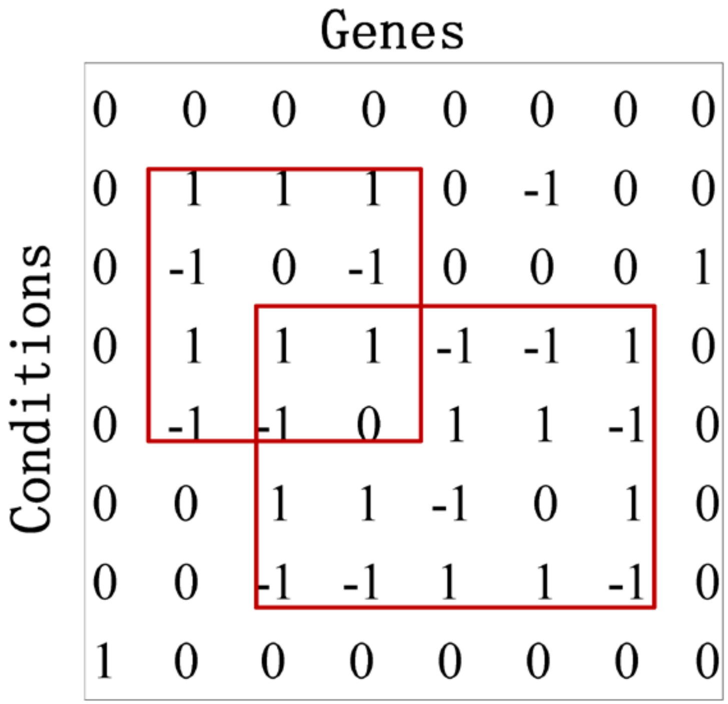
-1 down-regulated
0 unchanged
1 up-regulated

Biclustering Review

- Simultaneously group genes and conditions in a microarray

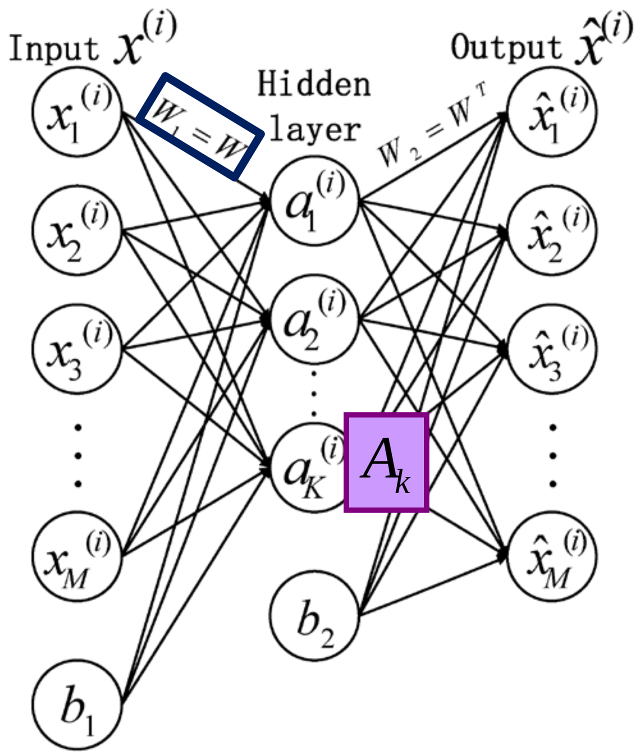
(Cheng and Church, ISMB'00)

- **Challenges:**
  - **Positive and negative correlation**
  - **Overlap in both genes and conditions**
  - **Not necessarily full coverage**
  - **Robustness against noise**

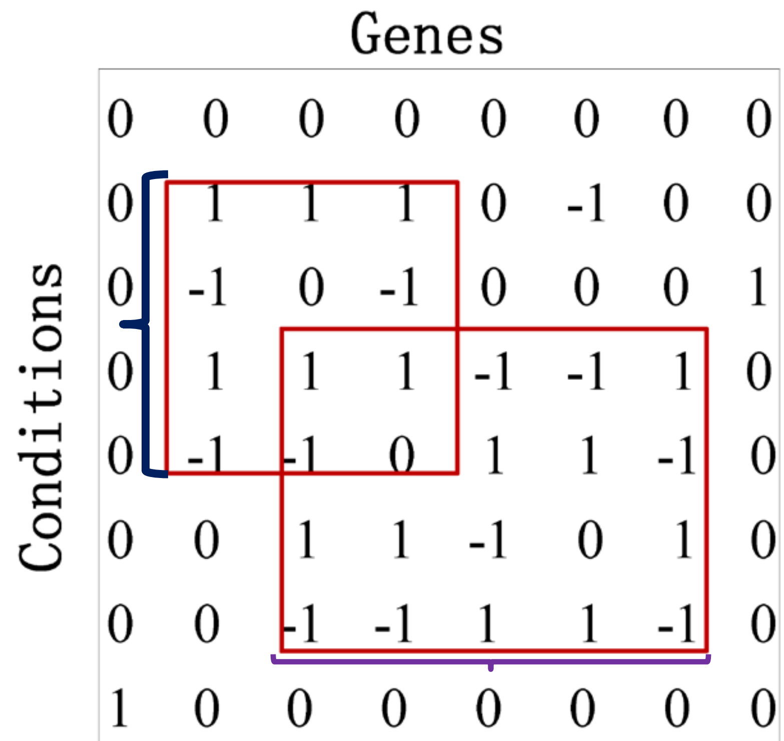


Map Sparse Autoencoder to Biclustering

Sparse Autoencoder (SAE)

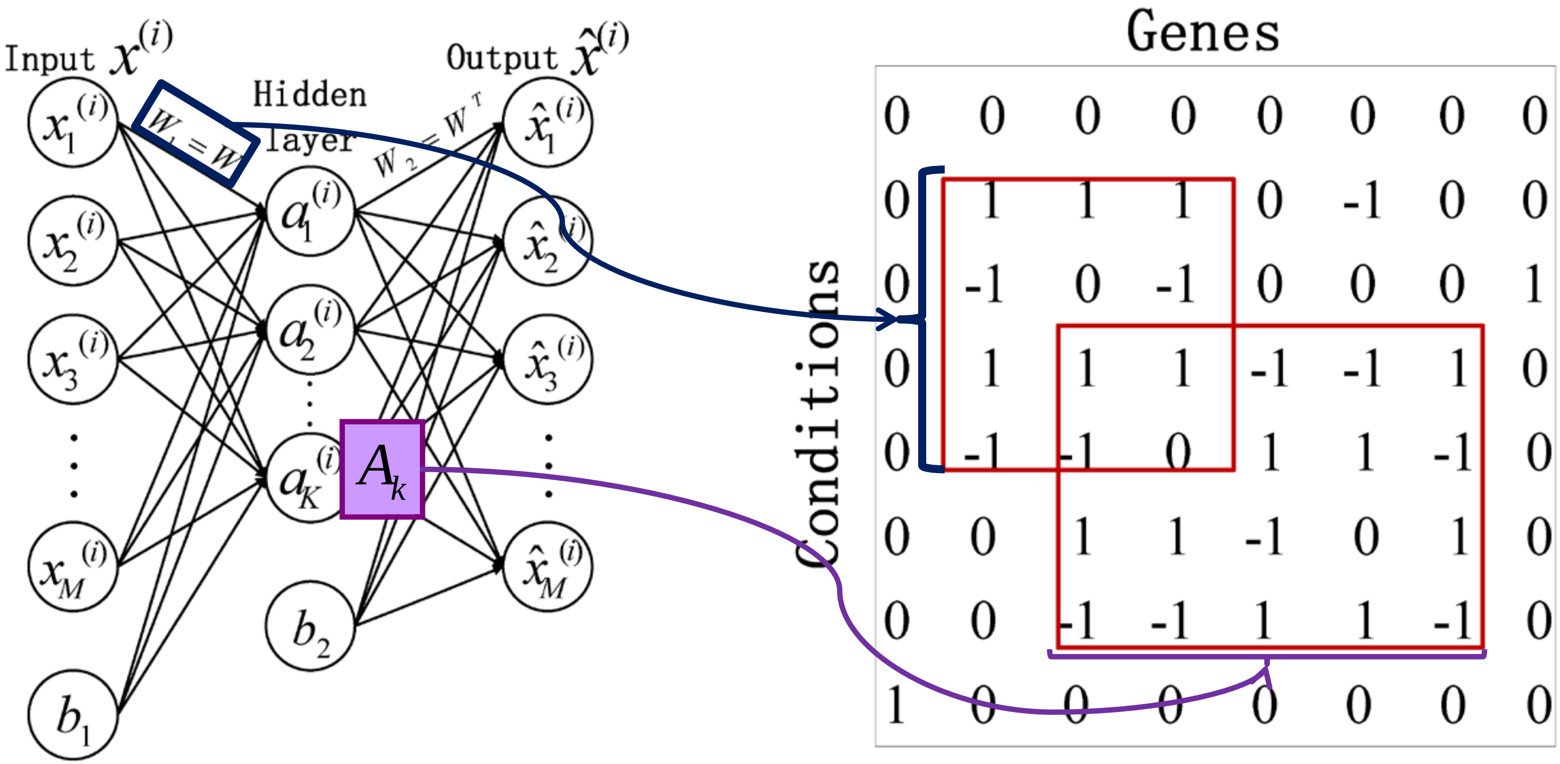


Biclustering



Map Sparse Autoencoder to Biclustering

One hidden neuron => one potential bicluster
W => membership of rows in biclusters
A => membership of columns in biclusters



Bicluster Embedding

For each hidden neuron *k*,

- Gene membership

  1. Pick N_k genes that have the largest N_k activation values into bicluster k, where $N_k = [N * \hat{\rho}_k]$;

  2. Among the selected N_k genes, remove those genes whose activation value is less than a threshold δ ($\delta \in (0,1)$) .

- Condition membership

  ➢ **Pick the m_{th} condition if $|W_{k,m}| > \xi$ $(\xi \in (0,1))$.**

Problems of Autoencoder

- Aim at "lowest reconstruction errors"

(recall $(\hat{x}_m^{(n)} - x_m^{(n)})^2$)

- However, we hope to ***capture patterns*** in noisy gene expression data

Genes

Conditions

0	0	0	1	0	0	1	0
0	1	1	1	0	1	0	0
-1	-1	0	-1	-1	0	0	0
0	1	1	1	1	1	0	0
0	0	0	0	-1	0	0	1

Original data

Genes

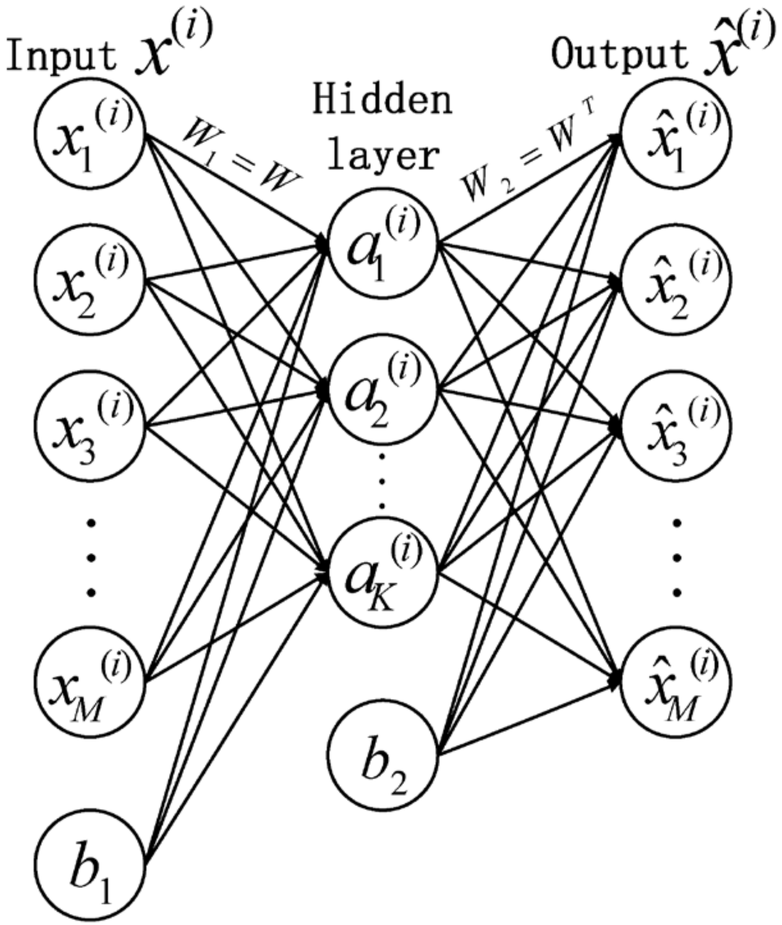
Conditions

0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0
0	-1	-1	-1	-1	-1	0	0
0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0

Patterns captured (desired)

Reconstruction error can be high.

Our Model: AutoDecoder (AD)



Optimization formulation

$$\underset{W,b_1,b_2}{argmin} \quad H = \frac{1}{2N} * \sum_{n=1}^{N} \sum_{m=1}^{M} [x_m^{(n)^2} * (\hat{x}_m^{(n)} - x_m^{(n)})^2$$

$$+ \beta_1 * (1 - x_m^{(n)^2}) * (\hat{x}_m^{(n)} - x_m^{(n)})^2] \quad (i)$$

$$+ \beta_2 * KL(\rho \| \hat{\rho}) \quad (ii)$$

$$+ \frac{\lambda}{2} * \|W - tanh(\eta * W)\|_F^2 \quad (iii)$$

Sparse Autoencoder (SAE) & AutoDecoder (AD)

SAE

$$\underset{W,b_1,b_2}{argmin} \quad H = \frac{1}{2N} * \sum_{n=1}^{N} \sum_{m=1}^{M} (\hat{x}_m^{(n)} - x_m^{(n)})^2 \quad (i)$$
$$+ \beta * KL(\rho \| \hat{\rho}) \quad (ii)$$
$$+ \frac{\lambda}{2} * \|W\|_F^2 \quad (iii)$$

AD

$$\underset{W,b_1,b_2}{argmin} \quad H = \frac{1}{2N} * \sum_{n=1}^{N} \sum_{m=1}^{M} [{x_m^{(n)}}^2 * (\hat{x}_m^{(n)} - x_m^{(n)})^2$$
$$+ \beta_1 * (1 - {x_m^{(n)}}^2) * (\hat{x}_m^{(n)} - x_m^{(n)})^2] \quad (i)$$
$$+ \beta_2 * KL(\rho \| \hat{\rho}) \quad (ii)$$
$$+ \frac{\lambda}{2} * \|W - tanh(\eta * W)\|_F^2 \quad (iii)$$

Improvement of AD over SAE:

(1) Term (*i*): non- uniform weighting

(2) Term(*iii*): weight polarization

Non-uniform Weighting (Term (i))

$$x_m^{(n)^2} * (\hat{x}_m^{(n)} - x_m^{(n)})^2 + \beta_1 * (1 - x_m^{(n)^2}) * (\hat{x}_m^{(n)} - x_m^{(n)})^2$$

- $\beta_1 > 1$ allows more false negative reconstruction errors.
- Tend to exclude non-zeros from final patterns than to include zeros inside the patterns.
- Resistance against Type A noise:

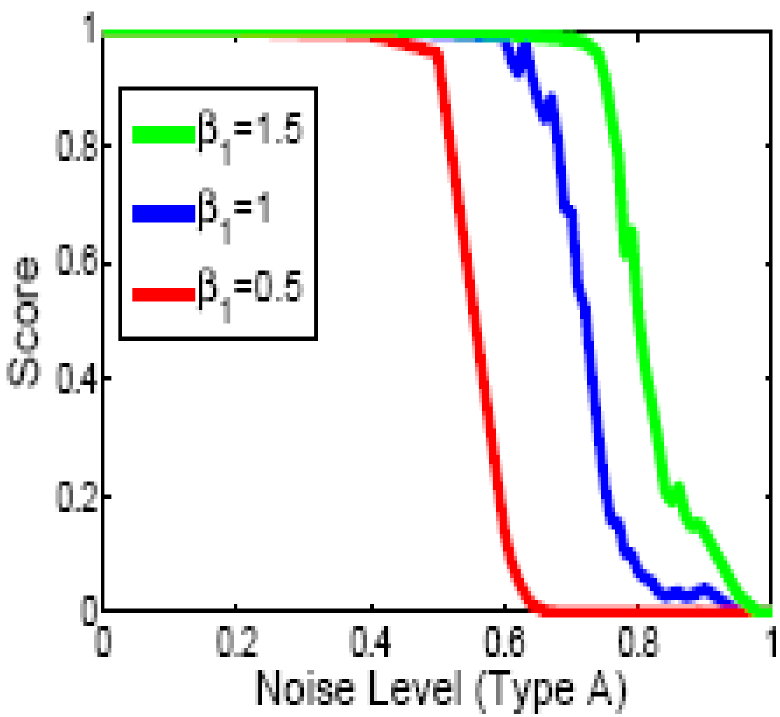
$$\begin{matrix} 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \end{matrix}$$

- $\beta_1 < 1$ allows more false positive reconstruction errors.
- Tend to include zeros inside final patterns than to exclude non-zeros from the patterns.
- Resistance against Type B noise:

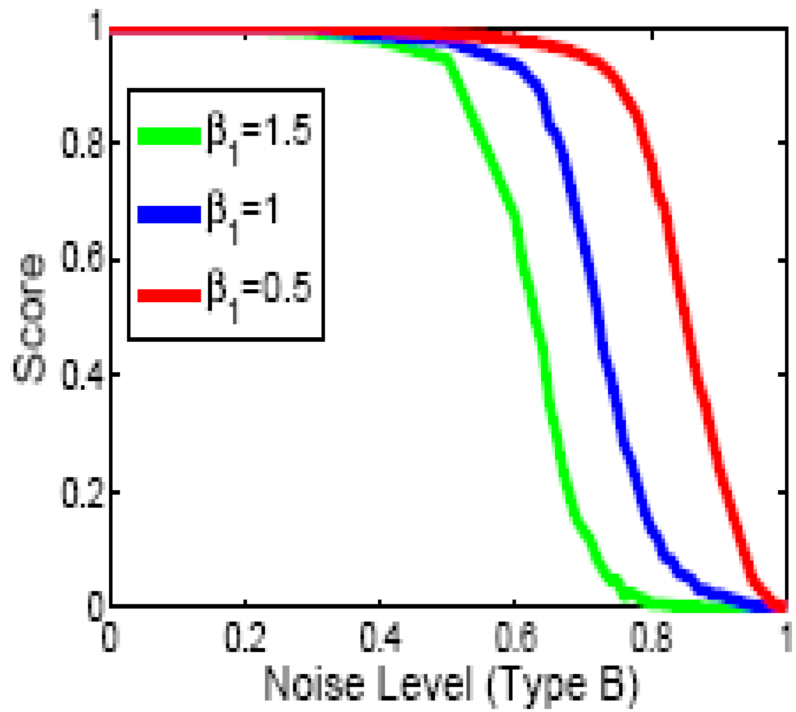
$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & -1 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Non-uniform Weighting (Term (i))

$\beta_1 > 1$: **Resistance to Type A noise**



$\beta_1 < 1$: **Resistance to Type B noise**

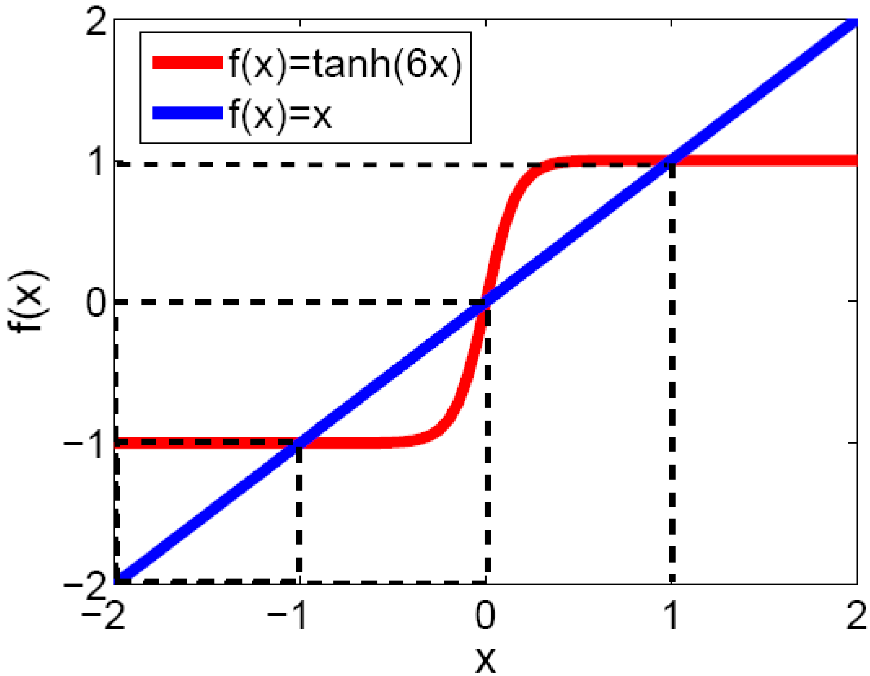


Weight Polarization (Term (iii))

$$\frac{\lambda}{2} * \|W - tanh(\eta * W)\|_F^2$$

- η can be any positive number *s.t.* the roots of $W - tanh(\eta * W) = 0$ appear at {-1, 0, 1} approximately.
- The threshold selection: more flexible in (0,1)

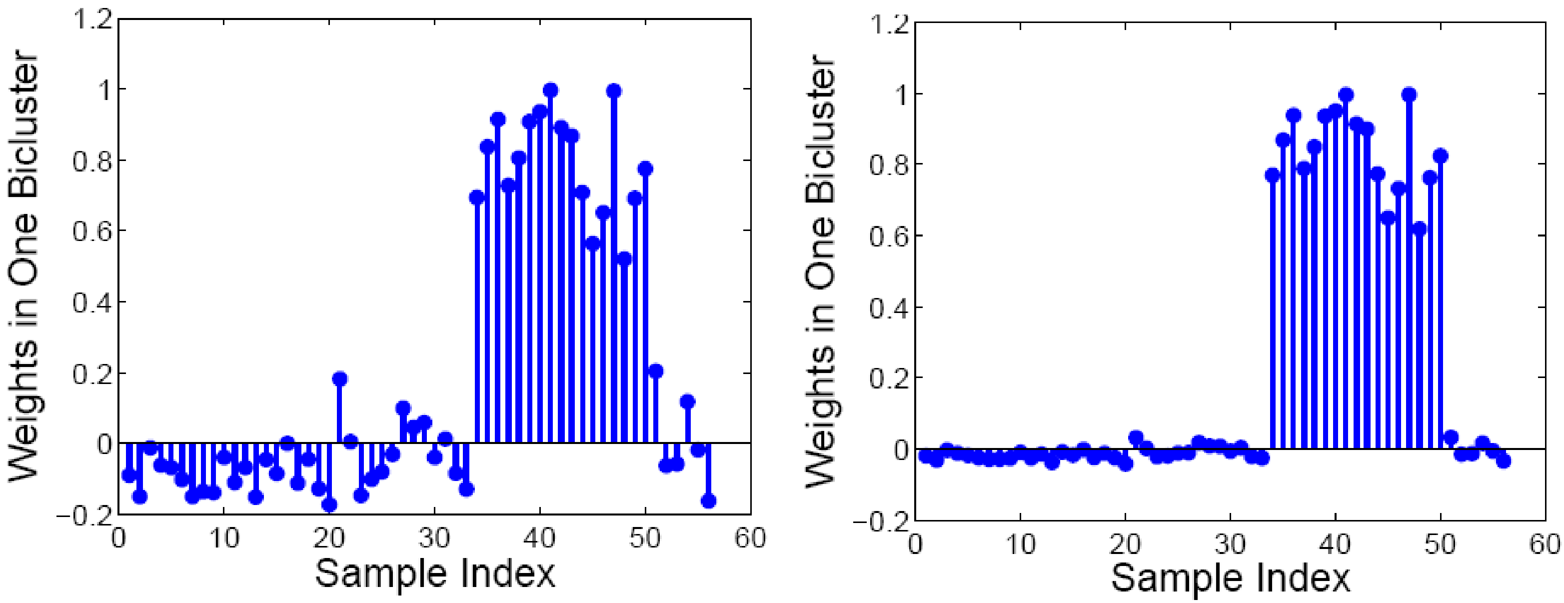
E.g. pick $\eta = 6$



Weight Polarization (Term (iii))

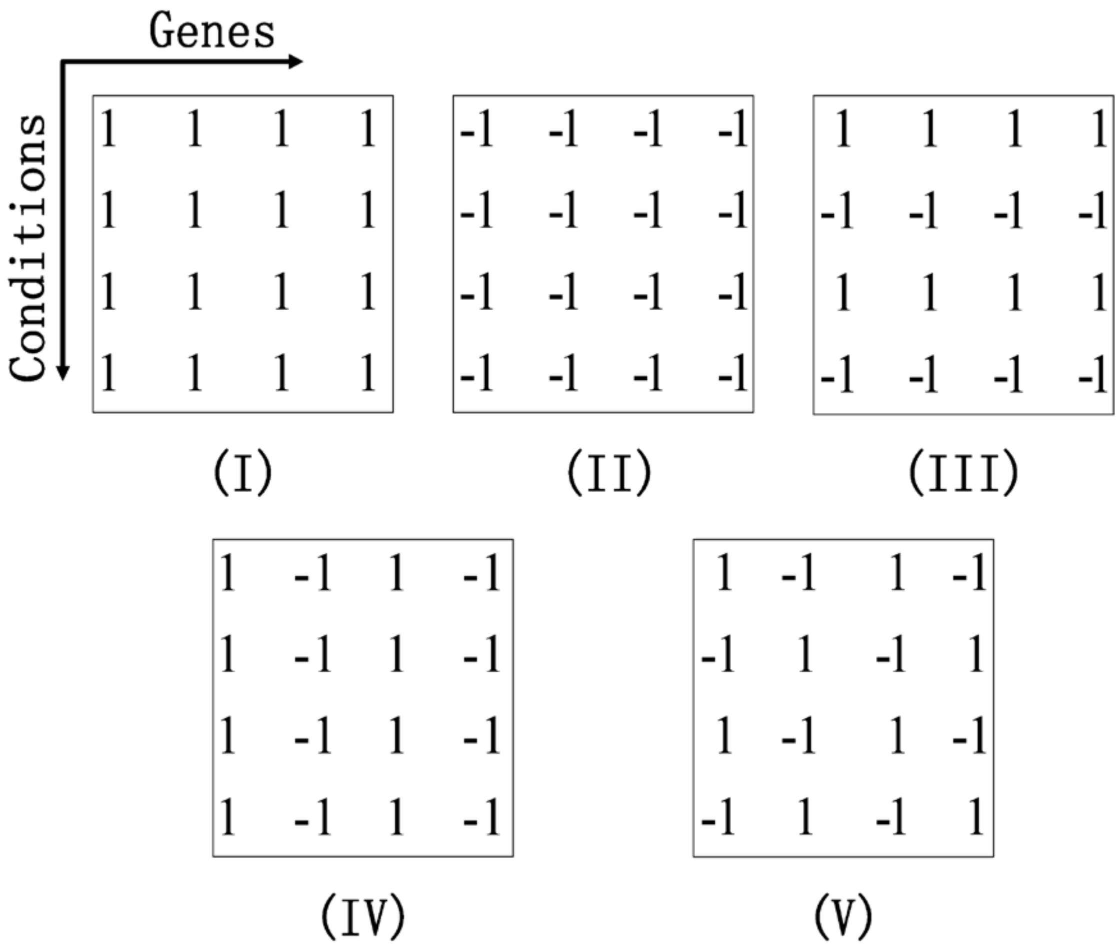
$$\frac{\lambda}{2} * \|W - tanh(\eta * W)\|_F^2$$

- η can be any positive number *s.t.* the roots of $W - tanh(\eta * W) = 0$ appear at {-1, 0, 1} approximately.
- The threshold selection: more flexible in (0,1)



One row of W learnt by $\frac{\lambda}{2} * \|W\|_F$ **(left) and** $\frac{\lambda}{2} * \|W - tanh(6 * W)\|_F$ **(right)**

Bicluster Patterns



(I-V) Readily captured by AD with an appropriate activation function in a hidden layer.

Outline

- **Introduction**
- A New Generation of Neural Networks
- Neural Networks & Biclustering
- Preliminary Results
- Future Work

Model Evaluation

- **Datasets (#g * #c)**

  Breast cancer (1213*97), multiple tissue (5565*102), DLBCL (3795*58), and lung cancer (12625*56).

- **Metric**
  - Relevance and recovery on condition sets
  - P-value analysis on gene sets

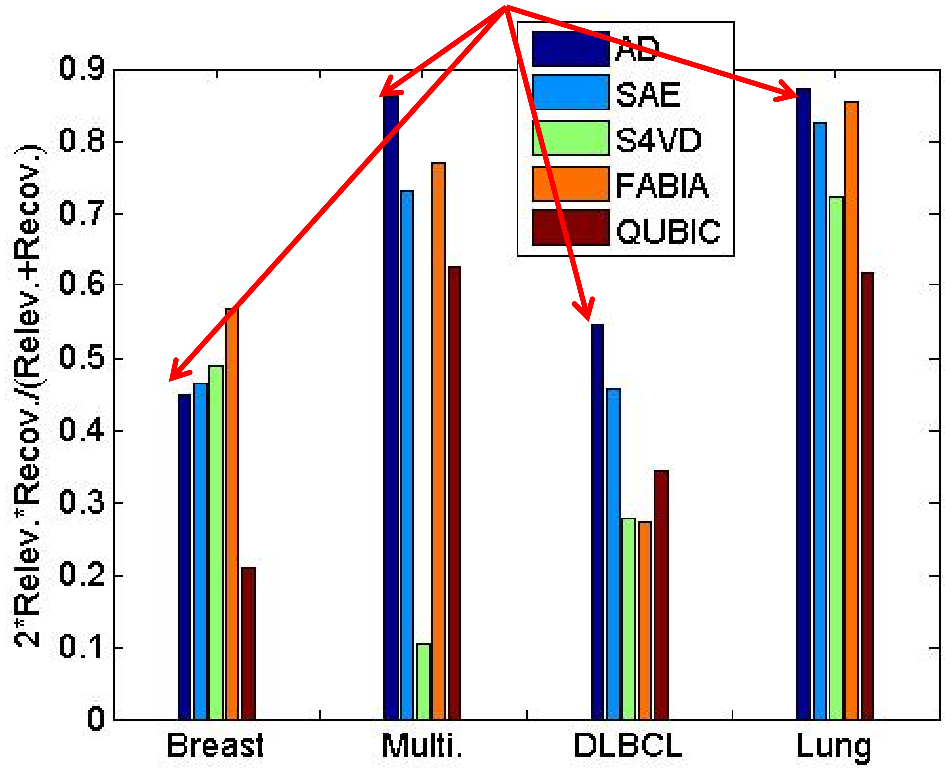
- **Comparison**
  - S4VD (matrix factorization approach, Bioinformatics'11)
  - FABIA (probabilistic approach, Bioinformatics'10)
  - QUBIC (combinatorial approach, NAR'09)

- **Environment**

  3.4GHZ, 16GB, Intel PC running Windows 7.

Experimental Results

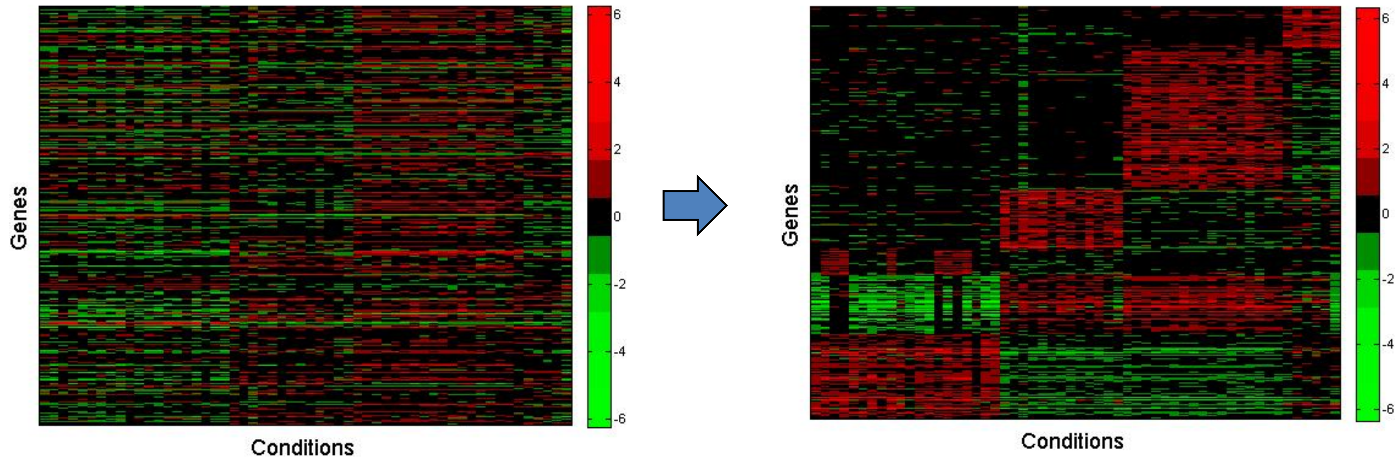
1. **Condition cluster evaluation by average relevance and recovery**



2. **Gene cluster evaluation by gene enrichment analysis**
   AD can generally discover biclusters with P-value less than 10^{-4} , much often less than 10^{-10} .

Experimental Results

Original lung cancer data **Biclusters discovered**



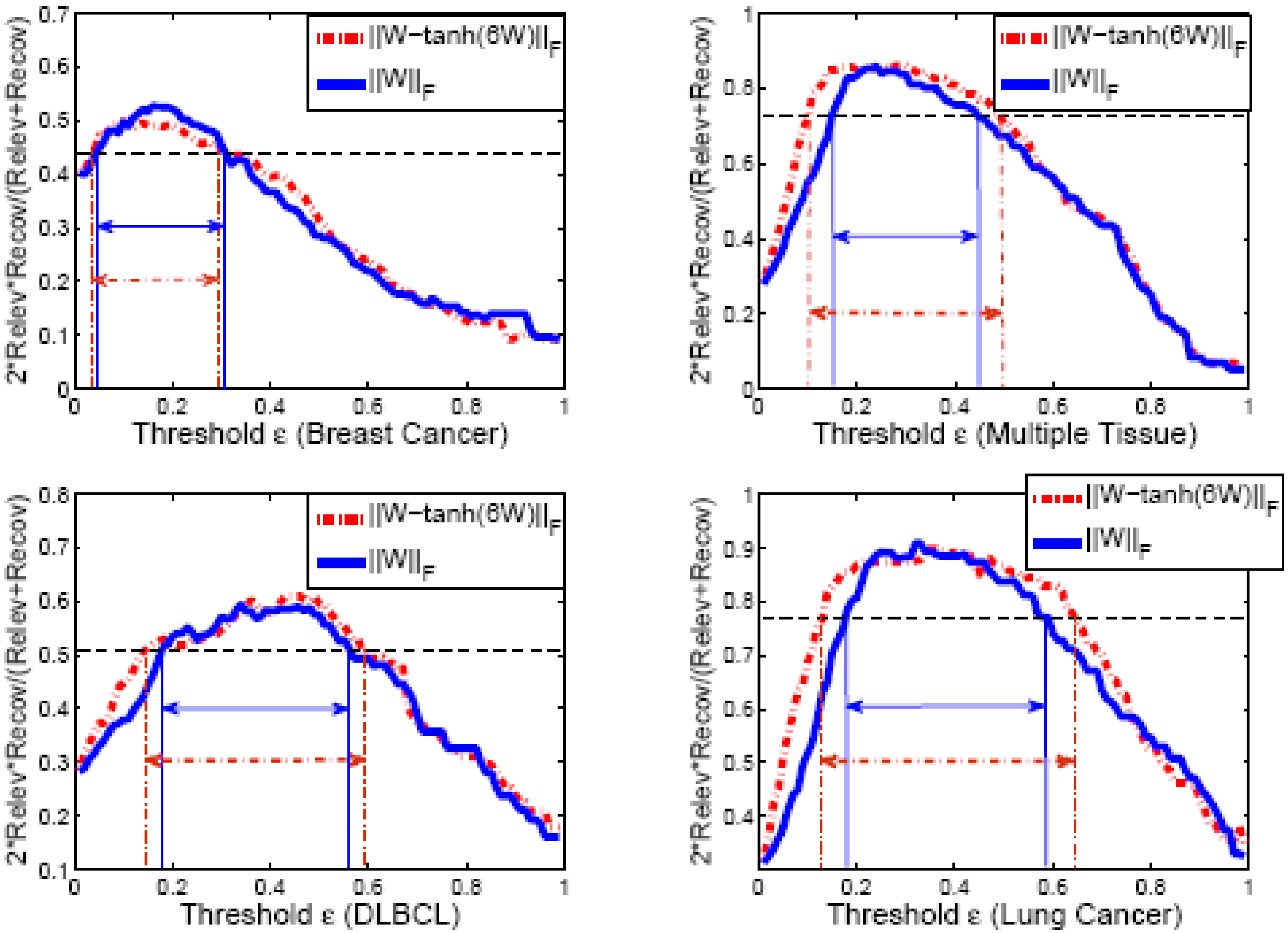
Conclusion:

1. **AutoDecoder guarantees the biological significance of the gene sets while improving the performance on condition sets.**

2. **AutoDecoder outperforms all the leading approaches that have been developed in the past 10 years.**

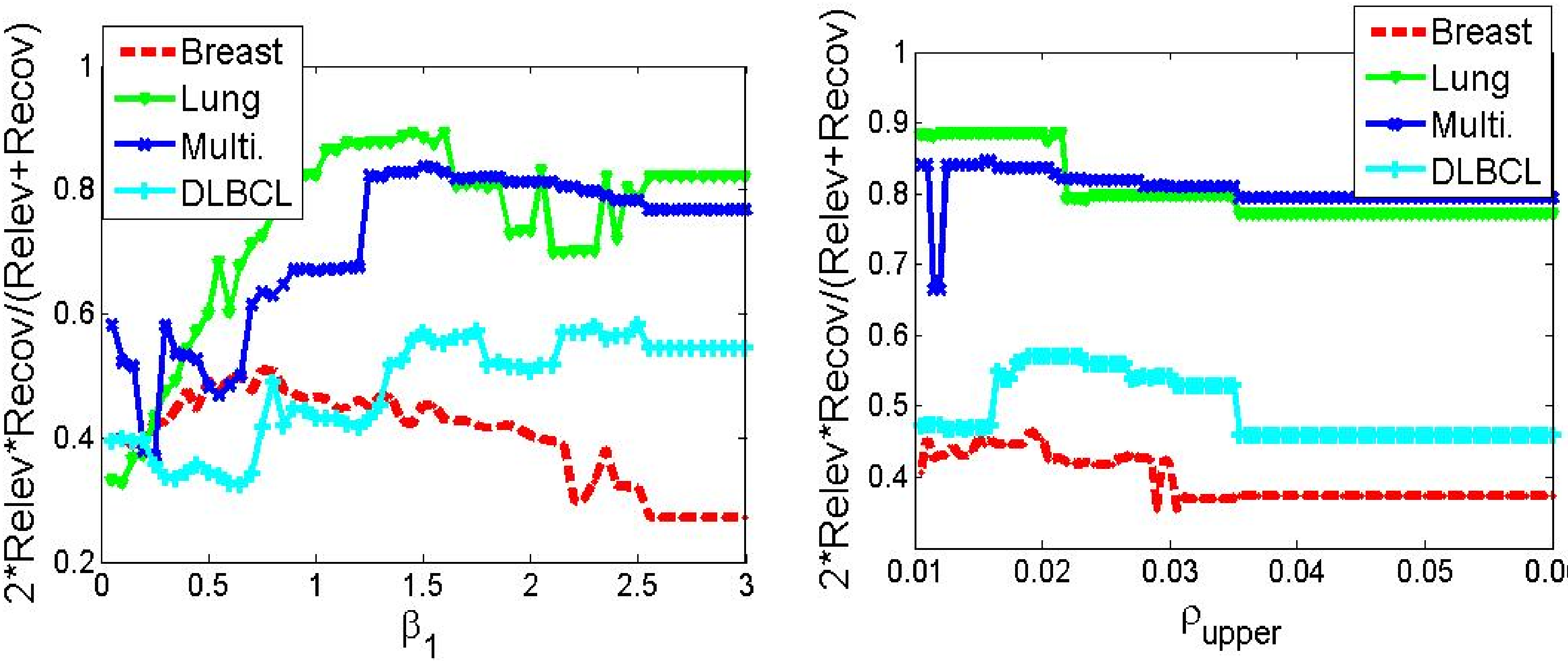
Parameter Sensitivity

- Condition Membership Threshold



Parameter Sensitivity

- Noise Resistant Parameter β_1 and activation rate $[\rho_{lower}, \rho_{upper}]$



Outline

- **Introduction**
- A New Generation of Neural Networks
- Neural Networks & Biclustering
- Preliminary Results
- Future Work

Future Work

- Apply neural networks outside text/vision/audio
  e.g. customers group mining

- Learn semantic features in text analysis to replace traditional language models

- Automatic text annotation for image segments

- Multiple object (unknown sizes) recognition in images

- Model robustness against noise (such as incorrect grammars, incomplete sentences, occlusion in images)

- …

References

[**1**] Hinton *et al*. Reducing the Dimensionality of Data with Neural Networks, Science, 2006;

[**2**] Bengio *et al*. Greedy Layer-Wise Training of Deep Networks, NIPS'07;

[3] Lee *et al*. Sparse Deep Belief Net Model for Visual Area V2, NIPS'08;

[4] Lee *et al*. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations, ICML'09;

[5] Socher *et al*. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions, EMNLP'11;

[**6**] Erhan *et al*. Why Does Unsupervised Pre-training Help Deep Learning? JMLR'10;

[7] Cheng *et al*. Biclustering of Gene Expression Data, ISMB'00;

[8] Mohamed *et al*. Acoustic Modeling Using Deep Belief Networks, IEEE Trans on Audio, Speech and Language Processing , 2012;

References

[9] Coates *et al*. An Analysis of Single-Layer Networks in Unsupervised Feature Learning, AISTATS'11;

[10] Socher *et al*. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection, NIPS'11;

[11] Goodfellow *et al*. Measuring Invariances in Deep Networks, NIPS'09;

[12] Socher *et al*. Parsing Natural Scenes and Natural Language with Recursive Neural Networks, ICML'11;

[13] Ranzato *et al*. On Deep Generative Models with Applications to Recognition, CVPR'11;

[14] Masci *et al*. Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction, ICANN'11;

[15] Raina *et al*. Self-taught Learning: Transfer Learning from Unlabeled Data, ICML'07;

Thank You !

Questions, please?