Machine learning for hearing research

Kang-Hun Ahn

Department of Physics Chungnam National University

I. **Introduction**
II. Hearing in noise: machine learning simulation
III. Speech recognition faster than FFT using machine learning
IV. Tinnitus as a neural net property
V. Unsupervised learning: Dictionary learning of sound

What Is Deep Learning?

Deep learning

From Wikipedia, the free encyclopedia

Deep learning (*deep machine learning*, or *deep structured learning*, or *hierarchical learning*, or sometimes *DL*) is a branch of machine learning based on a set of algorithms that attempt to model high-level abstractions in data by using model architectures, with complex structures or otherwise, composed of multiple non-linear transformations.[1](p198)[2][3][4]

REVIEW

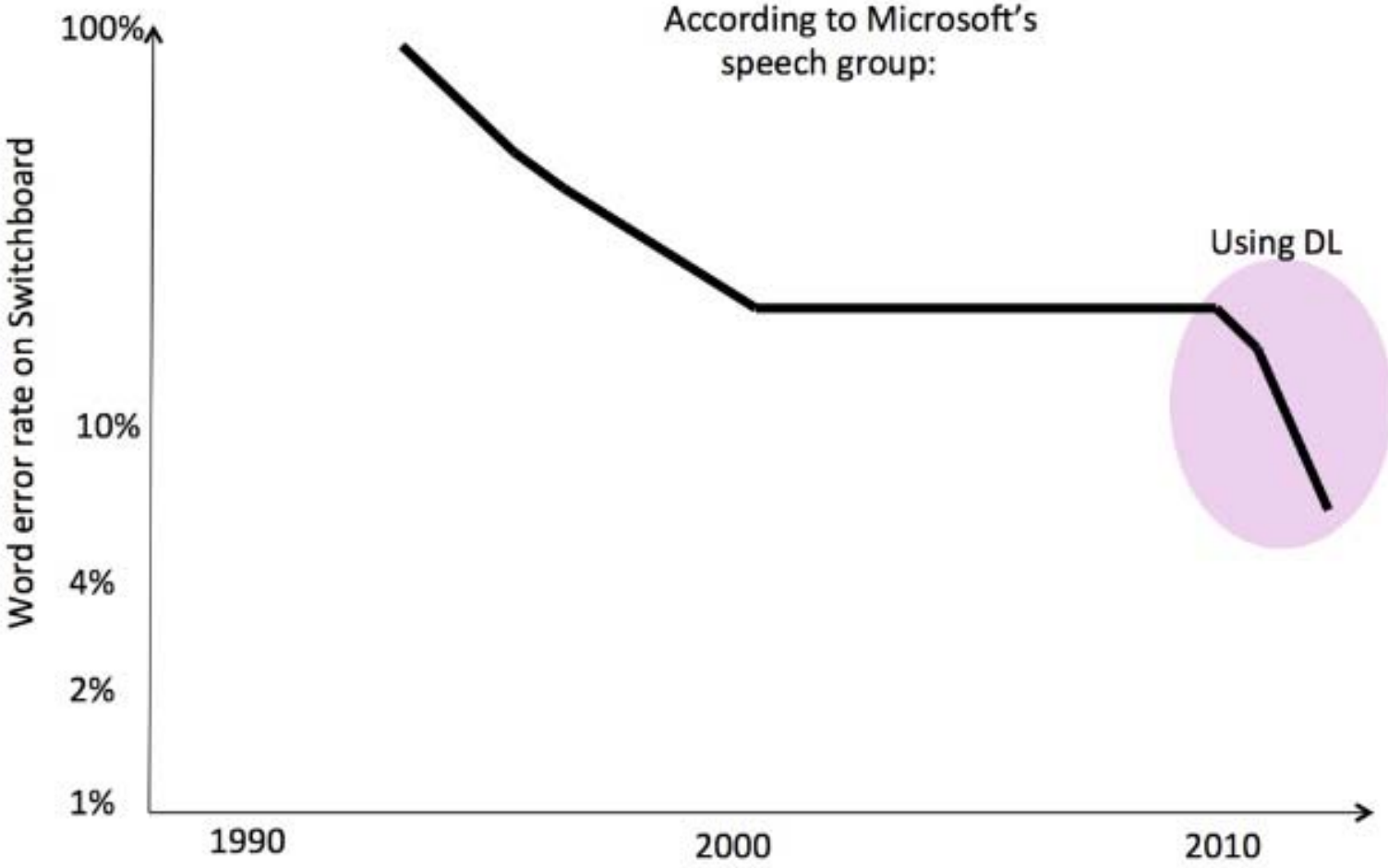
doi:10.1038/nature14539

Deep learning

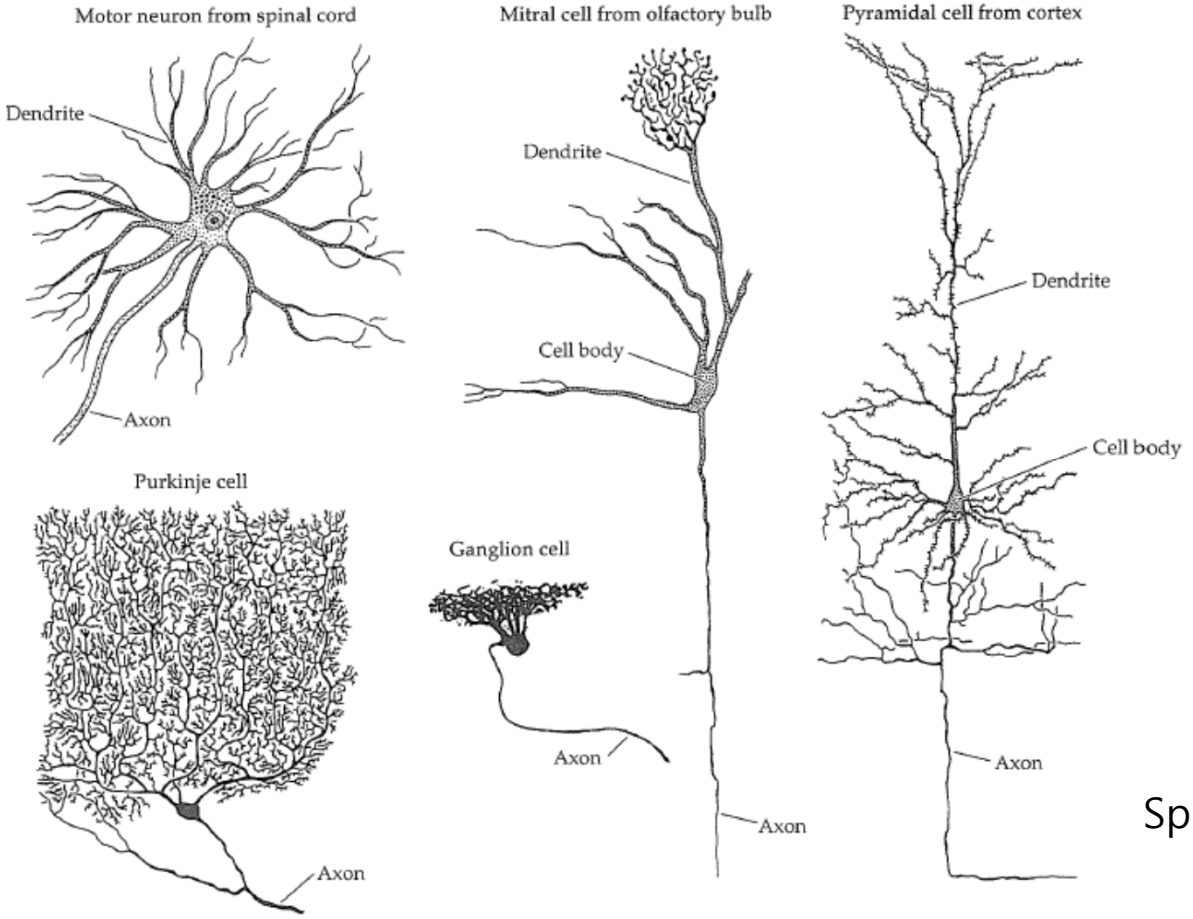
Yann LeCun[1,2], Yoshua Bengio[3] & Geoffrey Hinton[4,5]

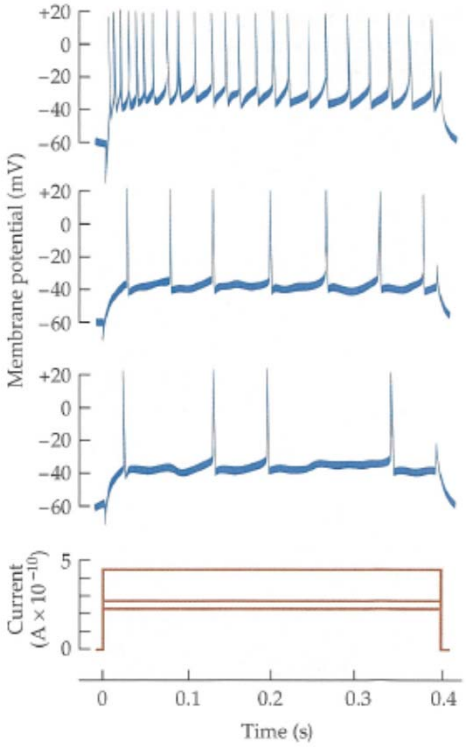
Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics. Deep learning discovers intricate structure in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer. Deep convolutional nets have brought about breakthroughs in processing images, video, speech and audio, whereas recurrent nets have shone light on sequential data such as text and speech.

Speech Recognition



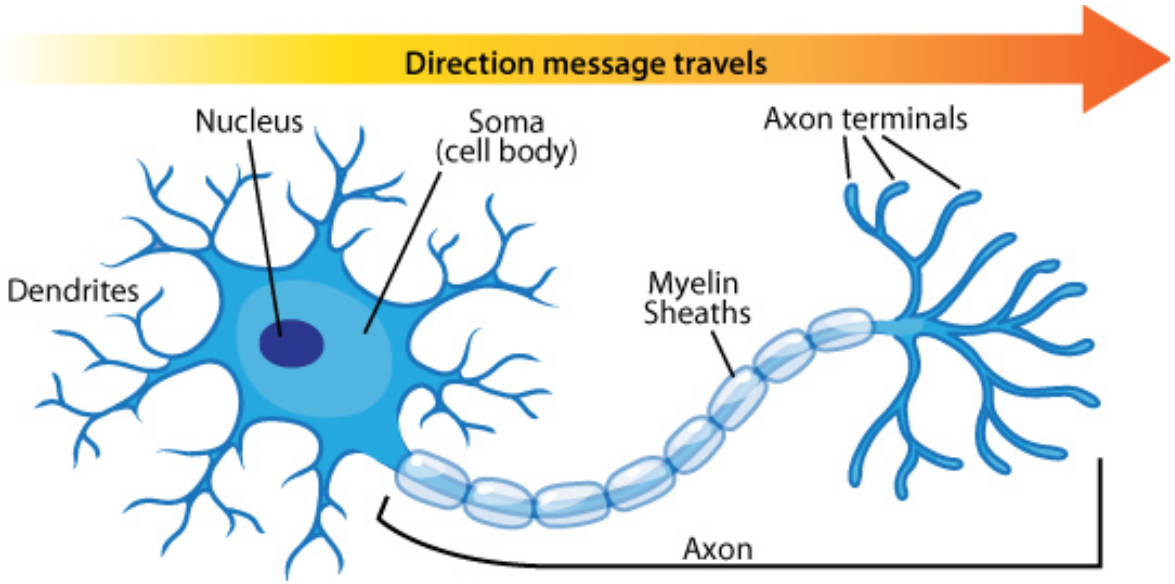
Biological neuron architecture



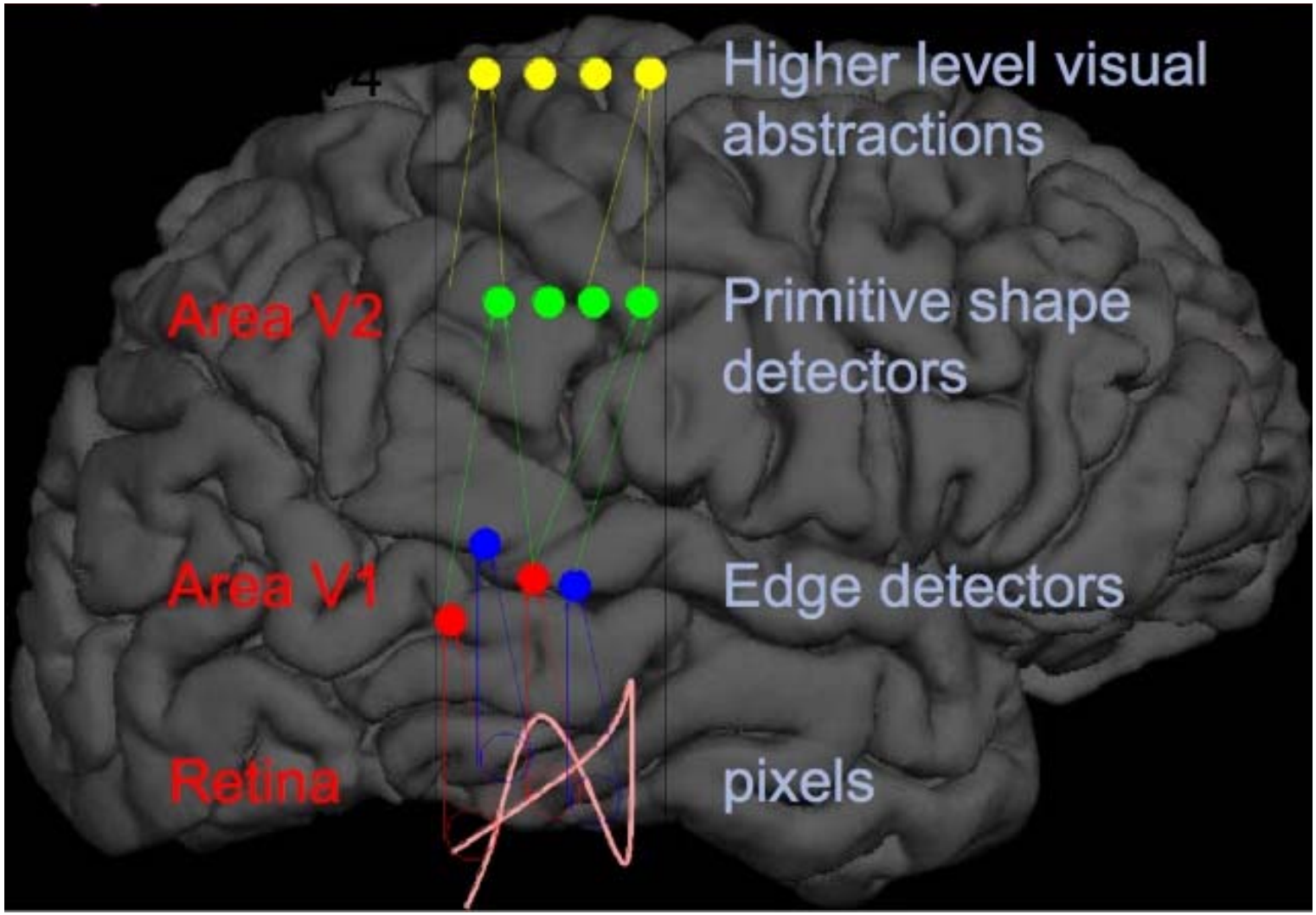


Spikes are generated above threshold stimulus

From neuron to brain



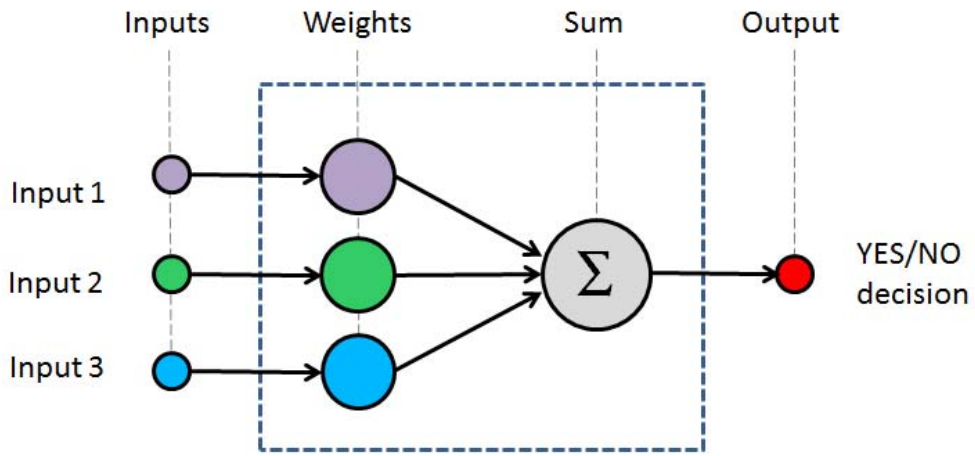
The information flow is one-way (directional).


Higher level visual abstractions
Area V2
Primitive shape detectors
Area V1
Edge detectors
Retina
pixels

Artificial Neuron

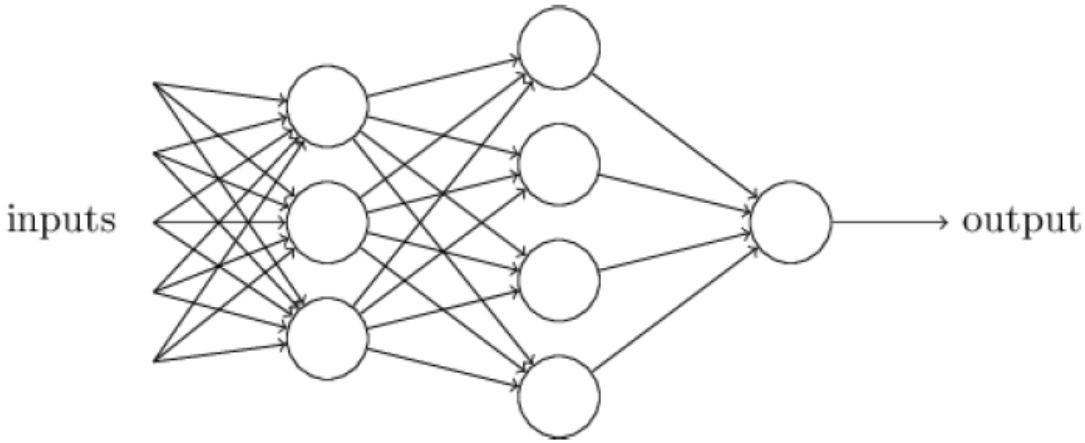
Perceptron

Frank Rosenblatt
(1927-1971)



$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

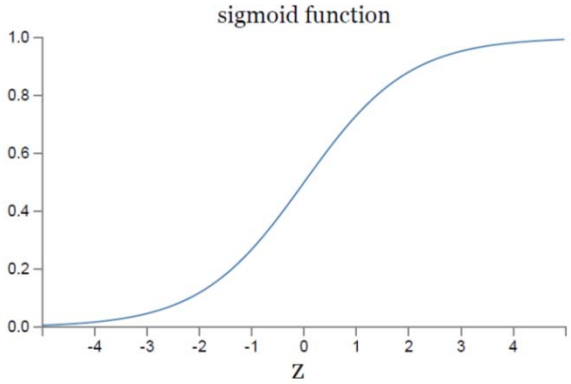
Layer of perceptrons



Problem: a small change in the weights or bias of any single perceptron can cause sometimes cause the output of that perceptron to completely flip, say 0 to 1.

Solution: **Sigmoid neuron**

$$\sigma(z) \equiv \frac{1}{1+e^{-z}}$$



Output = $\dfrac{1}{1+\exp(-\sum_j w_j x_j - b)}$

Types of Learning

Supervised (inductive) learning

- Training data includes desired outputs

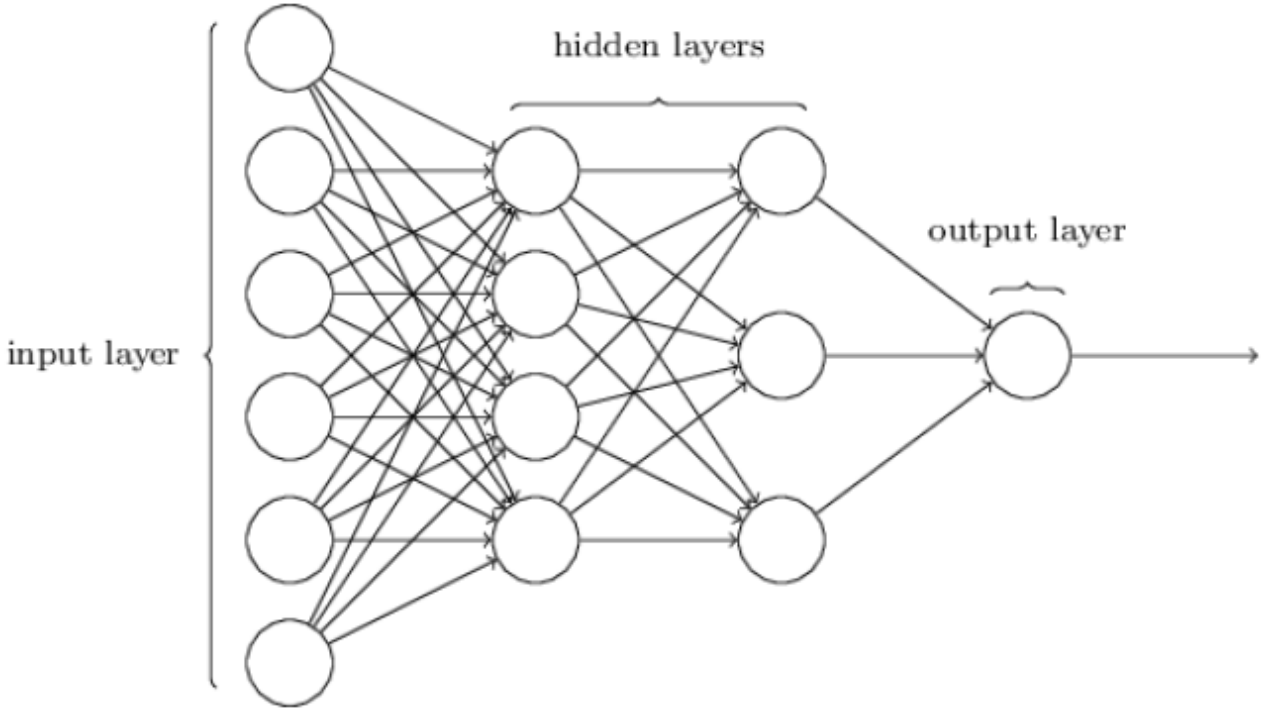
Unsupervised learning

- Training data does not include desired outputs

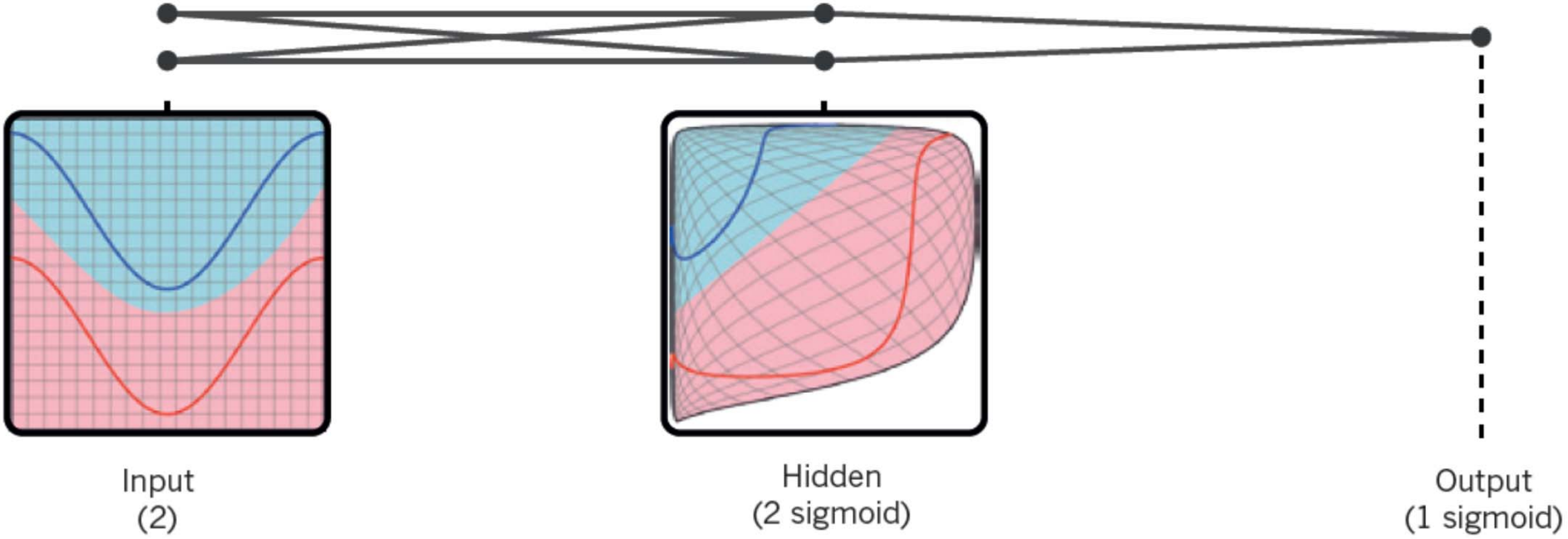
Semi-supervised learning

- Training data includes a few desired outputs

The architecture of neural network



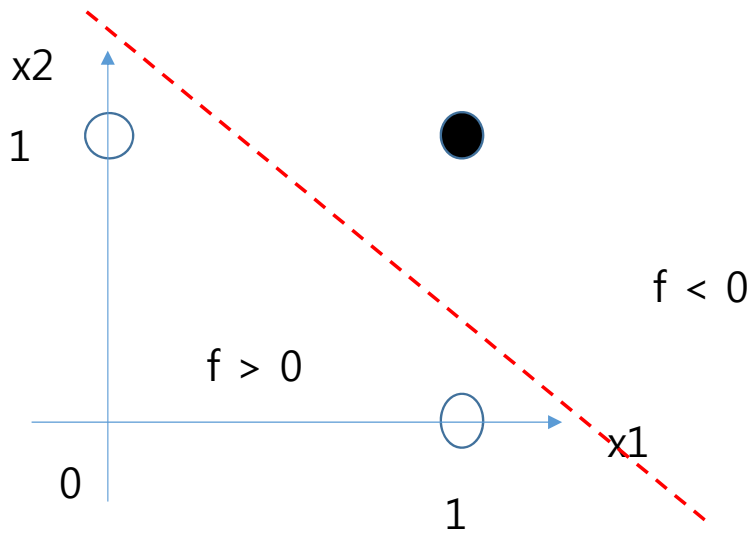
Role of hidden layer



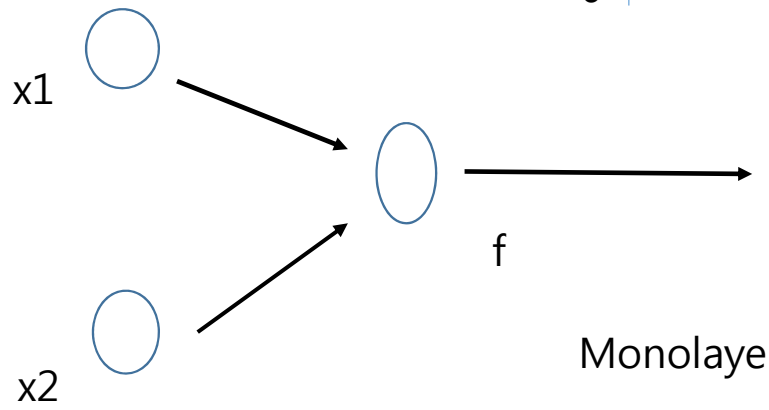
Example: AND operation

x1	x2	f
1	0	0
0	1	0
1	1	1

Desired output



Monolayer net is enough for AND operation.
One can find proper weight and bias.

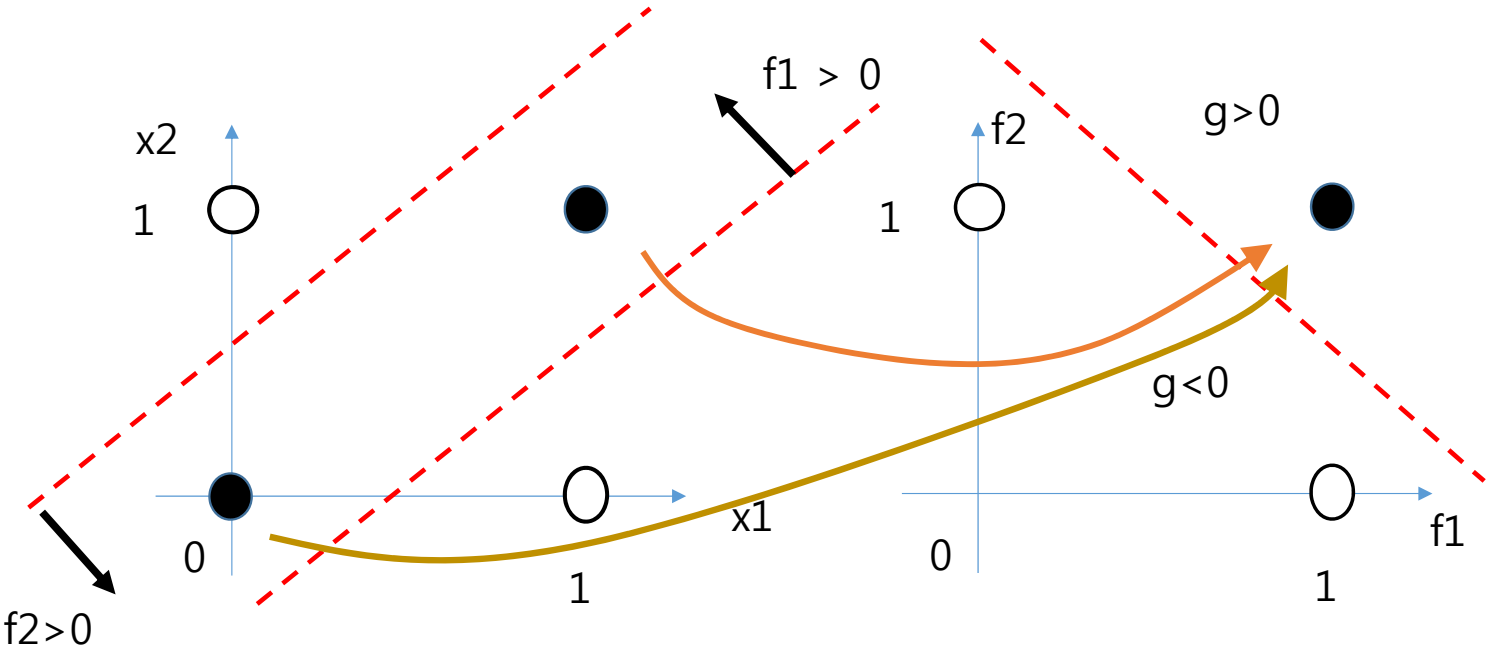


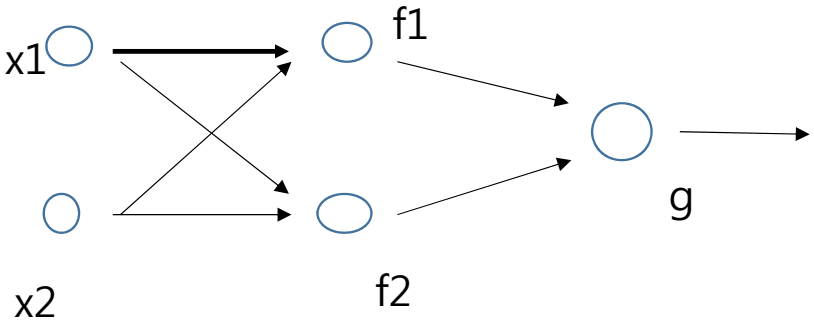
Monolayer net

$$f = w_{11}x_1 + w_{12}x_2 + b$$

XOR problem : Monolayer is not enough. Hidden layer is required.

x1	x2	f
1	0	0
0	1	0
1	1	1
0	0	1





f1=w11 x1 + w12 x2 +b1
f2= w21 x1 + w22 x2 +b2

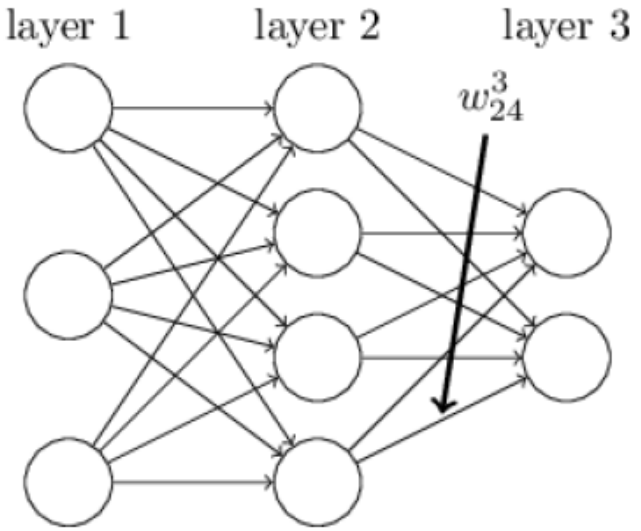
g=w'11 f1 + w'12 f2 +b'

Training : Minimizing an object function (error)

$$C(w,b) \equiv \sum_x \|y(x) - a\|^2$$

Desired output

Calculated output



$$a^l_j = \sigma\left(\sum_k w^l_{jk} a^{l-1}_k + b^l_j\right)$$

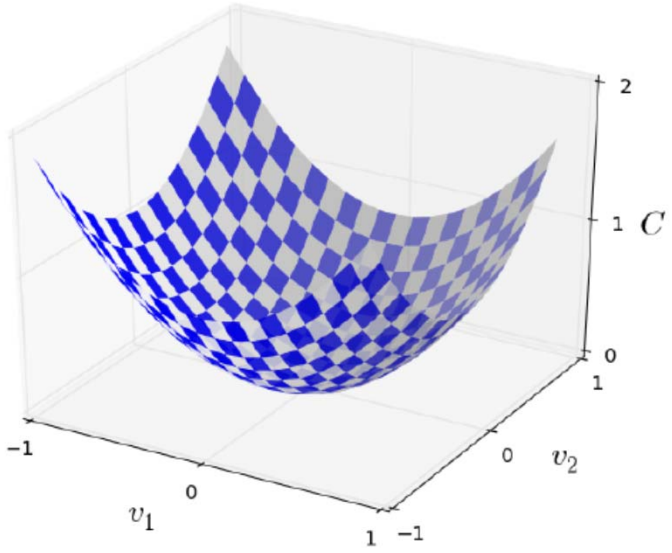
activation

Weight matrix

bias

Gradient descent method

$$\Delta C \approx \nabla C \cdot \Delta v.$$



$$\nabla C \equiv \left(\frac{\partial C}{\partial v_1}, \ldots, \frac{\partial C}{\partial v_m} \right)^T$$

$$\Delta v = (\Delta v_1, \ldots, \Delta v_m)^T$$

$$\Delta v = -\eta \nabla C.$$

Learning : Repeated application of

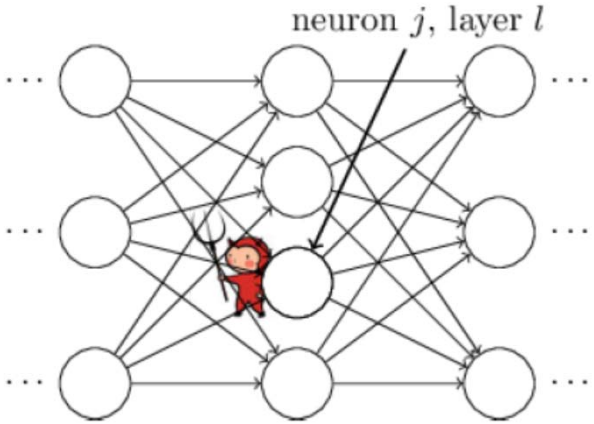
$$w_k \rightarrow w'_k = w_k - \eta \frac{\partial C}{\partial w_k}$$

$$b_l \rightarrow b'_l = b_l - \eta \frac{\partial C}{\partial b_l} .$$

Implementation:

1. **How to calculate the gradients ?**
2. **Batch learning, online learning ?**
3. **Learning rate η ?**
4. **Overtraining.**

The error of neuron δ_j^l



$$\delta_j^l \equiv \frac{\partial C}{\partial z_j^l} \qquad z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$$

,where the error cost function is

$$C = \frac{1}{2}\|y - a^L\|^2 = \frac{1}{2}\sum_j (y_j - a_j^L)^2,$$

The error of output layer neuron is clearly $\delta_j^L = \frac{\partial C}{\partial a_j^L}\sigma'(z_j^L)$

Back propagation

A simple linear equation for the error in terms of the error of next layer

$$\delta_j^l = \sum_k w_{kj}^{l+1} \delta_k^{l+1} \sigma'(z_j^l)$$

Proof)

$$\begin{aligned}
\delta_j^l &= \frac{\partial C}{\partial z_j^l} \\
&= \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} \\
&= \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l} \delta_k^{l+1},
\end{aligned}$$

where

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} a_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1} \sigma(z_j^l) + b_k^{l+1}.$$

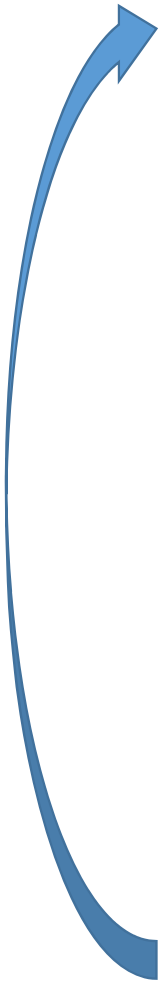
$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1} \sigma'(z_j^l)$$

1. How to calculate the gradients?
Solution:

The equations of backpropagation

1. $\delta^L = \nabla_a C \odot \sigma'(z^L)$
2. $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$
3. $\frac{\partial C}{\partial b_j^l} = \delta_j^l$
4. $\frac{\partial \mathrm{C}}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l$

Hardamard product



1. **Input x:** Set the corresponding activation a^1 for the input layer.

2. **Feedforward:** For each $l = 2, 3, \ldots, L$ compute $z^l = w^l a^{l-1} + b^l$ and $a^l = \sigma(z^l)$.

3. **Output error δ^L:** Compute the vector $\delta^L = \nabla_a C \odot \sigma'(z^L)$.

4. **Backpropagate the error:** For each $l = L-1, L-2, \ldots, 2$ compute $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$.

5. **Output:** The gradient of the cost function is given by $\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j$ and $\frac{\partial C}{\partial b^l_j} = \delta^l_j$.

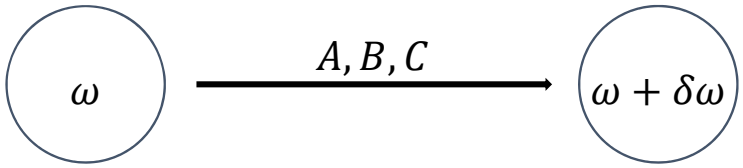
6. **Gradient descent:** For each $l = L, L-1, \ldots, 2$ update the weights according to the rule $w^l \to w^l - \frac{\eta}{m} \sum_x \delta^{x,l} (a^{x,l-1})^T$, and the biases according to the rule $b^l \to b^l - \frac{\eta}{m} \sum_x \delta^{x,l}$.

2. Batch learning or online learning?

Suppose you have different data sets A,B, C

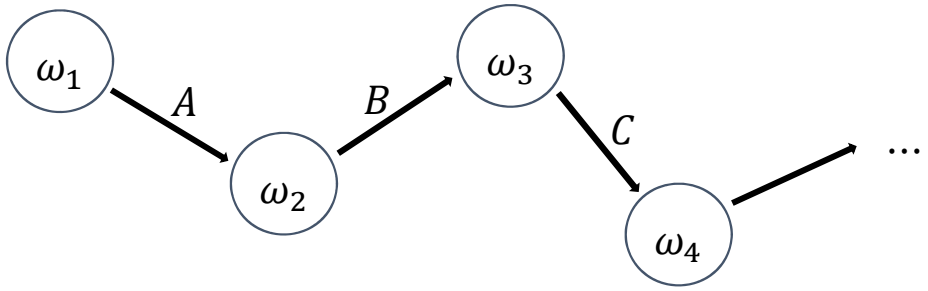
· Batched Learning

$$\delta\omega = -\eta \sum_{\alpha=A,B,C} \left\langle \alpha \left| \frac{\partial C(\omega)}{\partial \omega} \right| \alpha \right\rangle$$



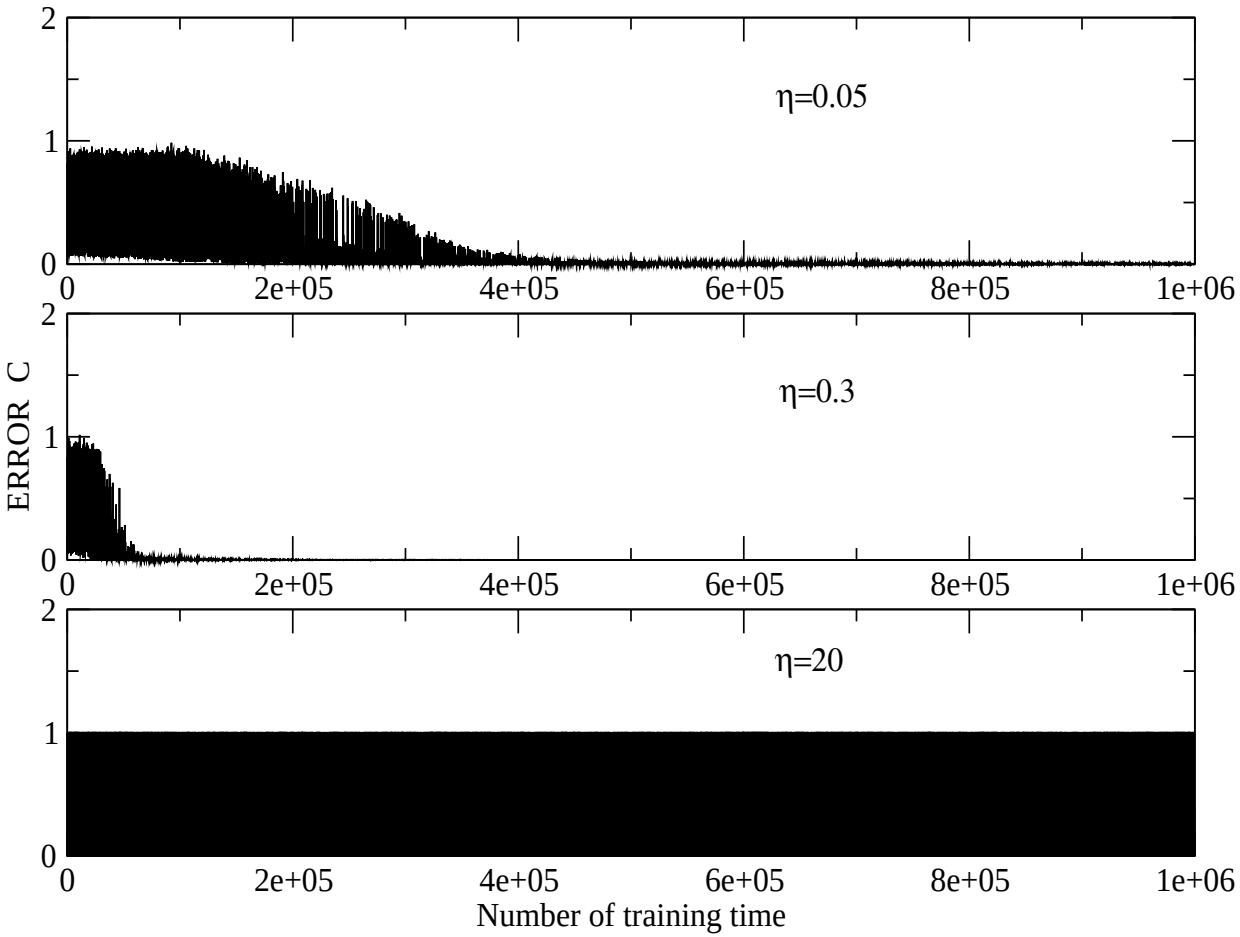
· Online Learning

$$\delta\omega = -\eta \left\langle A \left| \frac{\partial C(\omega_1)}{\partial \omega} \right| A \right\rangle - \eta \left\langle B \left| \frac{\partial C(\omega_2)}{\partial \omega} \right| B \right\rangle - \cdots$$



2. Batched learning or online learning?

- These two processes are mathematically different.
- Both of them work well for training.



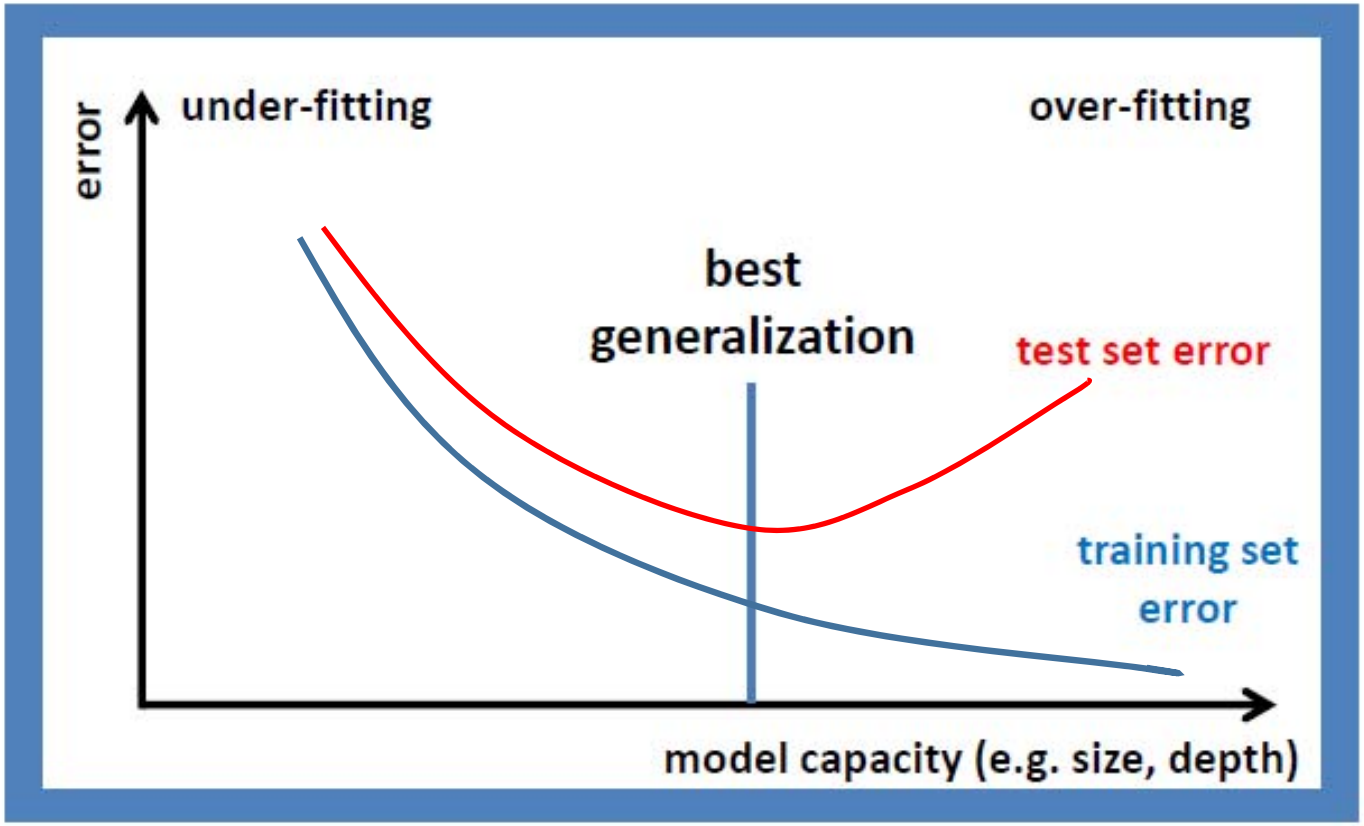
Too small learning rate gives too slow convergence.

Too large learning rate does not give convergence.

There exists an range of optimal learning rate which must be found through experience.

4. Overtraining

Too much training will hurt you ~~

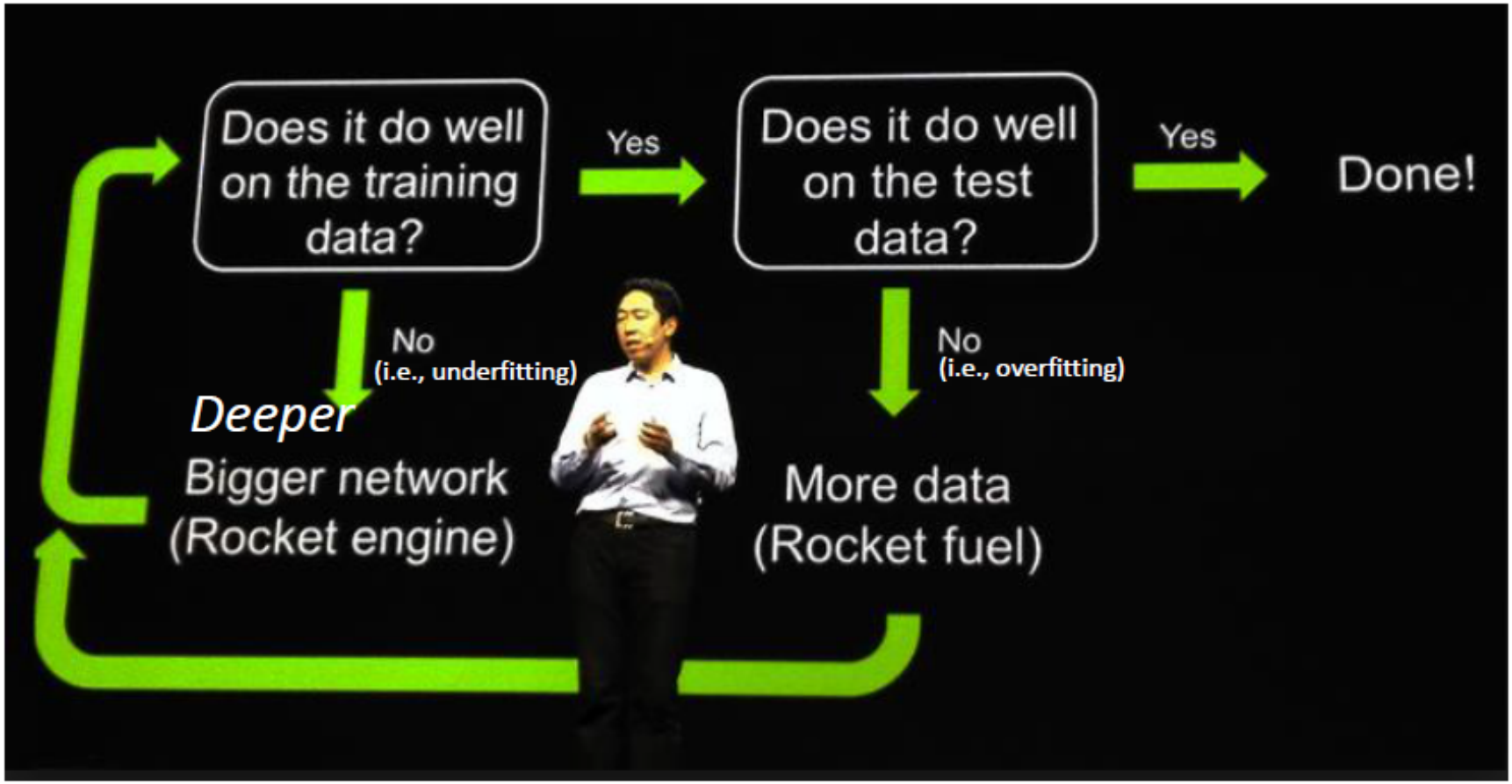


Most people in this field know this phenomenon but there is no mathematical proof.


"I HAVE NEVER SEEN FOOD THAT IS NOT CIRCLE-SHAPED. ALL FOOD IS THEREFORE DEFINITELY CIRCLE-SHAPED."
"I HAVE NEVER SEEN FOOD THAT DOES NOT HAVE STRAIGHT EDGES..."

Microsoft Research

Recipe for (Supervised) Deep Learning with Big Data



Technical problems

Why gradients tend to vanish for DNN

- To illustrate to problem, let's use matrix form of error BP:

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$$

$$= ((w^{l+2})^T \cdot \left[((w^{l+2})^T \delta^{l+2}) \odot \sigma'(z^{l+1}) \right]) \odot \sigma'(z^{l+1})$$

$$\ldots, \ldots, \ldots$$

This problem becomes more serious as the layer becomes deeper.

- So even if forward pass in nonlinear, the error backprop is a linear process
- It suffers from all problems associated with linear processes
- Many terms of σ (1-σ)
- In addition, many terms in the product of W's
- If any sigmoid unit saturates in either direction, the error gradient becomes zero
- If ||W||<1, the product will shrink fast for high depths
- If ||W||>1, the product may grow fast for high depths

From Le Deng's lecture note

1. $\delta^L = \nabla_a C \odot \sigma'(z^L)$
2. $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$
3. $\frac{\partial C}{\partial b_j^l} = \delta_j^l$
4. $\frac{\partial \mathrm{C}}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l$

Smallness of σ' slow down the learning process

variations

Log-likehood cost function

$$C = -\ln a_y^L$$

This is useful to avoid the slow-down problem as

$$\frac{\partial C}{\partial w_{jk}^L} = a_k^{L-1}(a_j^L - y_j)$$

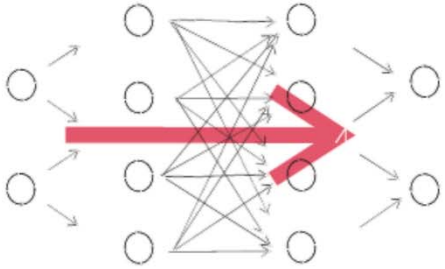
$$\frac{\partial C}{\partial b_j^L} = a_j^L - y_j$$

Softmax outlayer

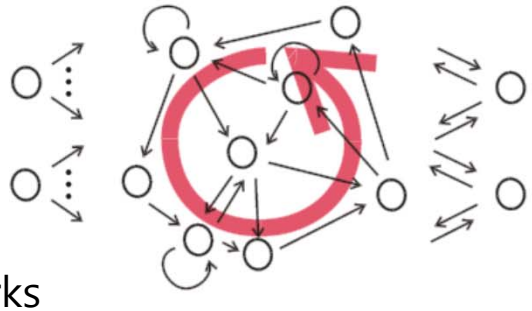
$$a_j^L = \frac{e^{z_j^L}}{\sum_k e^{z_k^L}}$$

Can be interpreted as probability of the output j as sum of a_j^L is equal to 1.

Feedfoward network



Recurrent network



Feedfowrd networks

-mathematically they implement static input-output mapping

Multi-layer perception(MLP) can approximate arbitrary nonlinear functions with arbitrary precision

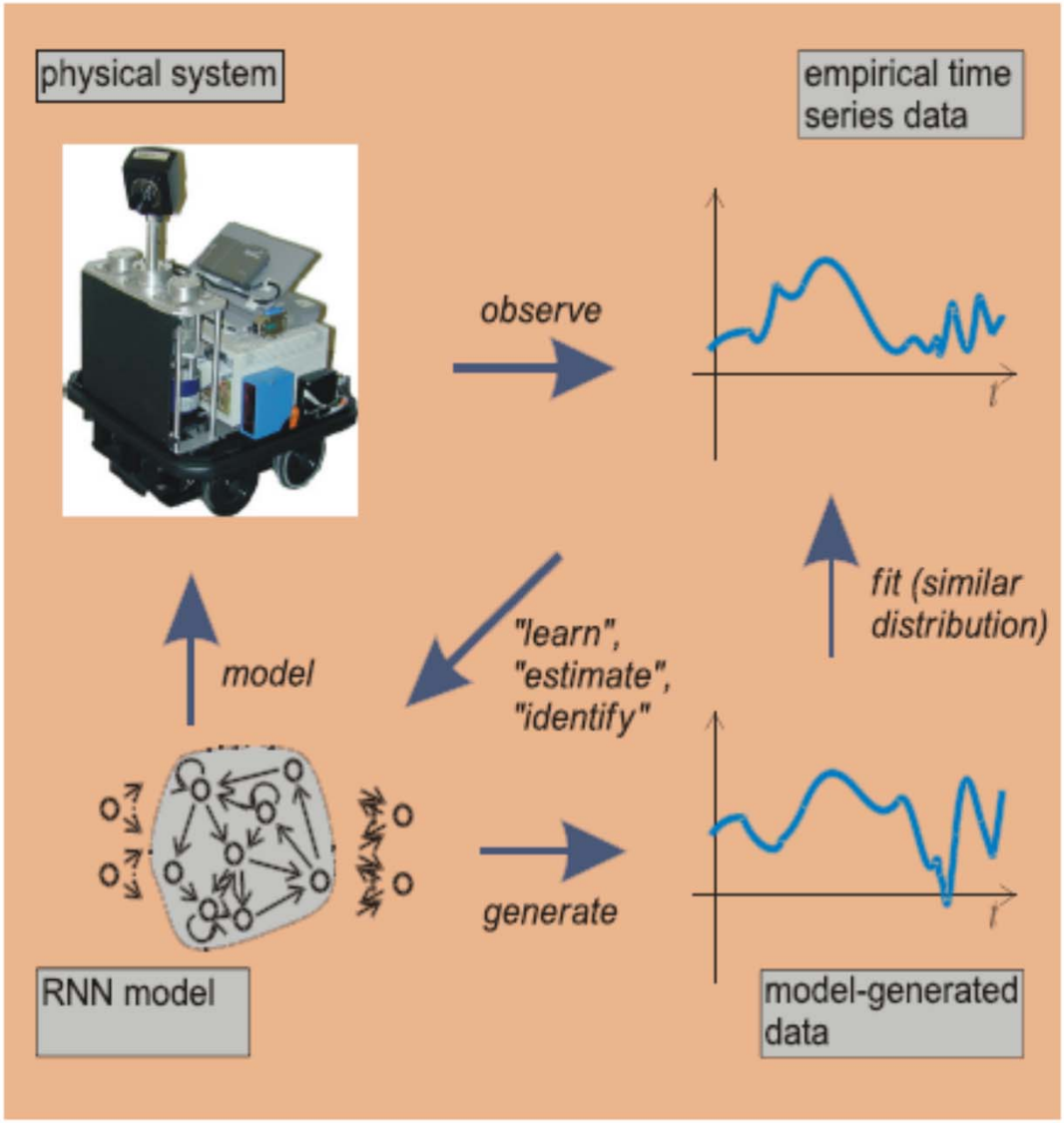
-Most popular supervised training algorithm: backpropagation algorithm

-Most (95%?) of neural network publication concern feedforward net

-have proven useful in many practical applications as pattern classifications

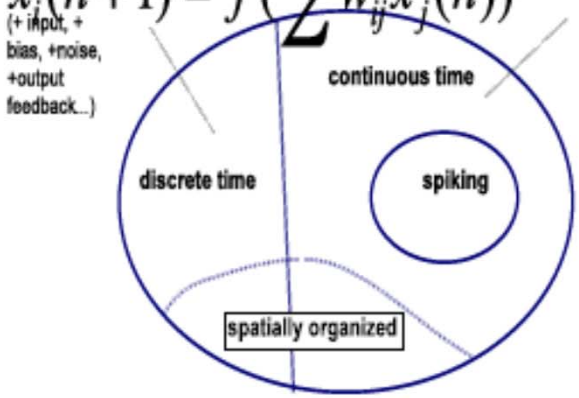
Recurrent network

- all biological neural networks are recurrent
- mathematically, RNNs implement dynamical systems
- basic theoretical result: RNNs can approximate arbitrary (term needs some qualification) dynamical systems with arbitrary precision ("universal approximation property")
- several types of training algorithms are known, no clear winner
- theoretical and practical difficulties by and large have prevented practical applications so far


physical system
empirical time series data
observe
t
fit (similar distribution)
"learn", "estimate", "identify"
model
generate
t
RNN model
model-generated data

Formal description of RNN

$$x_i(n+1) = f\left(\sum w_{ij} x_j(n)\right)$$



$$\tau \dot{x}_i = -x_i + \sum w_{ij} f(x_j)$$

n=1,2,3,... denotes the time

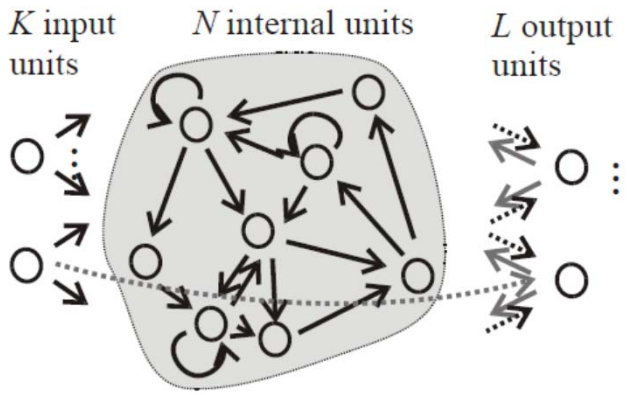
K input units $\mathbf{u}(n) = (u_1(n), \ldots, u_K(n))^t$

N internal units $\mathbf{x}(n) = (x_1(n), \ldots, x_N(n))^t$

L output units $\mathbf{y}(n) = (y_1(n), \ldots, y_L(n))^t$

$$\mathbf{W}^{in} = (w_{ij}^{in}), \quad \mathbf{W} = (w_{ij}), \quad \mathbf{W}^{out} = (w_{ij}^{out}) \quad \mathbf{W}^{back} = (w_{ij}^{back}).$$

N X K N X N L X (N+K) N X L



Updates of internal units

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{W}^{in}\mathbf{u}(n+1) + \mathbf{W}\mathbf{x}(n) + \mathbf{W}^{back}\mathbf{y}(n))$$

Input internal output

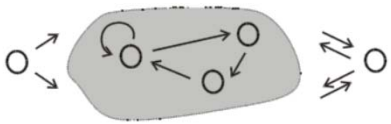
The output

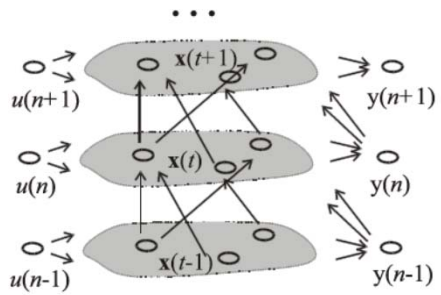
$$\mathbf{y}(n+1) = \mathbf{f}^{out}(\mathbf{W}^{out}(\mathbf{u}(n+1), \mathbf{x}(n+1)).$$

f= tanh or 1

Training of recurrent network

Backpropagation through time (BPTT) method





Unfold the recurrent network in time, by stacking identical copies of RNN !

Teacher data $\mathbf{u}(n) = (u_1(n),\ldots,u_K(n))'$, $\mathbf{d}(n) = (d_1(n),\ldots,d_L(n))'$ $n = 1,\ldots T$

The error to be minimized $E = \sum_{n=1,\ldots,T} \|\mathbf{d}(n) - \mathbf{y}(n)\|^2 = \sum_{n=1,\ldots,T} E(n)$

where

$$\mathbf{y}(n+1) = \mathbf{f}^{out}(\mathbf{W}^{out}(\mathbf{u}(n+1),\mathbf{x}(n+1)),$$

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{W}^{in}\mathbf{u}(n+1) + \mathbf{W}\mathbf{x}(n) + \mathbf{W}^{back}\mathbf{y}(n)),$$

Then the algorithm is straightforward feedforward backpropagation algorithm.

$$\delta_j(T) = (d_j(T) - y_j(T)) \frac{\partial f(u)}{\partial u}\bigg|_{u=z_j(T)}$$ error for the output units

$$\delta_i(T) = \left[\sum_{j=1}^{L} \delta_j(T) w_{ji}^{out}\right] \frac{\partial f(u)}{\partial u}\bigg|_{u=z_i(n)}$$ error for internal units x_i (t)

$$\delta_j(n) = \left[(d_j(n) - y_j(n)) + \sum_{i=1}^{N} \delta_i(n+1) w_{ij}^{back}\right] \frac{\partial f(u)}{\partial u}\bigg|_{u=z_j(n)}$$ error for the output units of earlier layer

$$\delta_i(n) = \left[\sum_{j=1}^{N} \delta_j(n+1) w_{ji} + \sum_{j=1}^{L} \delta_j(n) w_{ji}^{out}\right] \frac{\partial f(u)}{\partial u}\bigg|_{u=z_i(n)}$$ error for the internal units of earlier times

Adjustment

$$new\ w_{ij} = w_{ij} + \gamma \sum_{n=1}^{T} \delta_i(n)\, x_j(n-1) \quad [use\ x_j(n-1) = 0\ for\ n = 1]$$

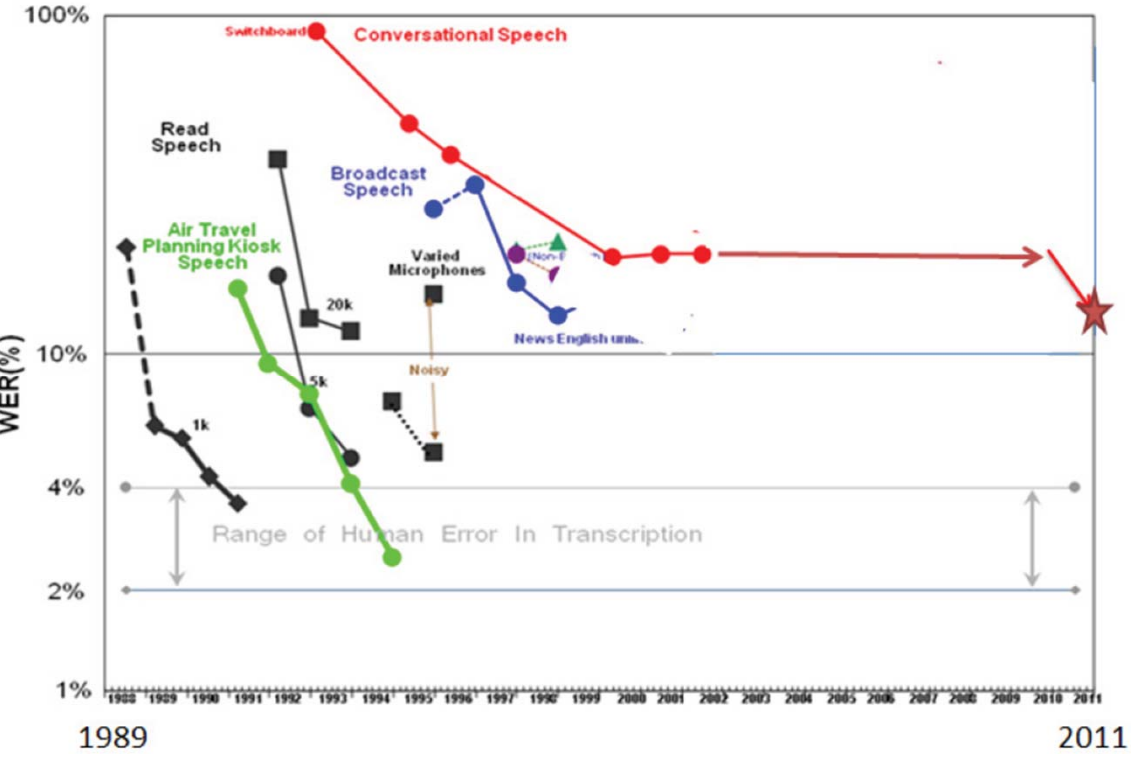
$$new\ w_{ij}^{in} = w_{ij}^{in} + \gamma \sum_{n=1}^{T} \delta_i(n)\, u_j(n)$$

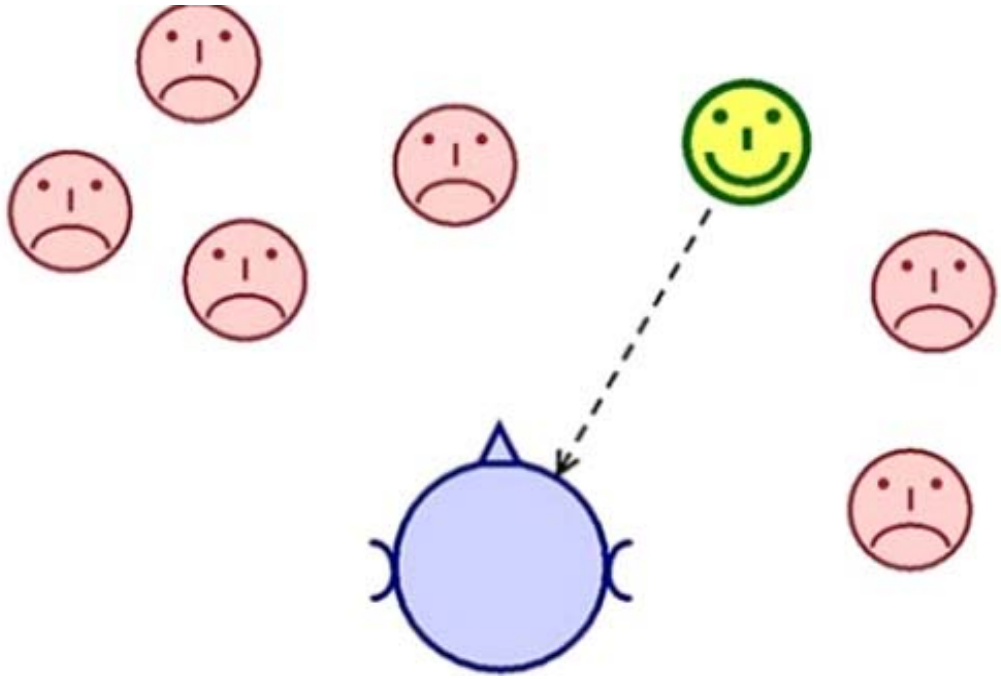
$$new\ w_{ij}^{out} = w_{ij}^{out} + \gamma \times \begin{cases} \sum_{n=1}^{T} \delta_i(n)\, u_j(n), if\ \ j\ refers\ to\ input\ unit \\ \sum_{n=1}^{T} \delta_i(n)\, x_j(n), if\ \ j\ refers\ to\ hidden\ unit \end{cases}$$

$$new\ w_{ij}^{back} = w_{ij}^{back} + \gamma \sum_{n=1}^{T} \delta_i(n)\, y_j(n-1) \quad [use\ y_j(n-1) = 0\ for\ n = 1]$$

I. Introduction
II. Hearing in noise: machine learning simulation
III. Speech recognition faster than FFT using machine learning
IV. Tinnitus as a neural net property
V. Unsupervised learning: Dictionary learning of sound

How humans can understand speech in spite of background noise is a key issue in auditory science

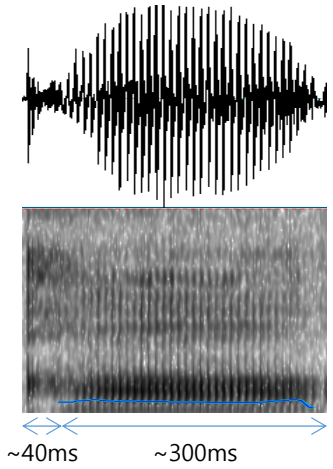




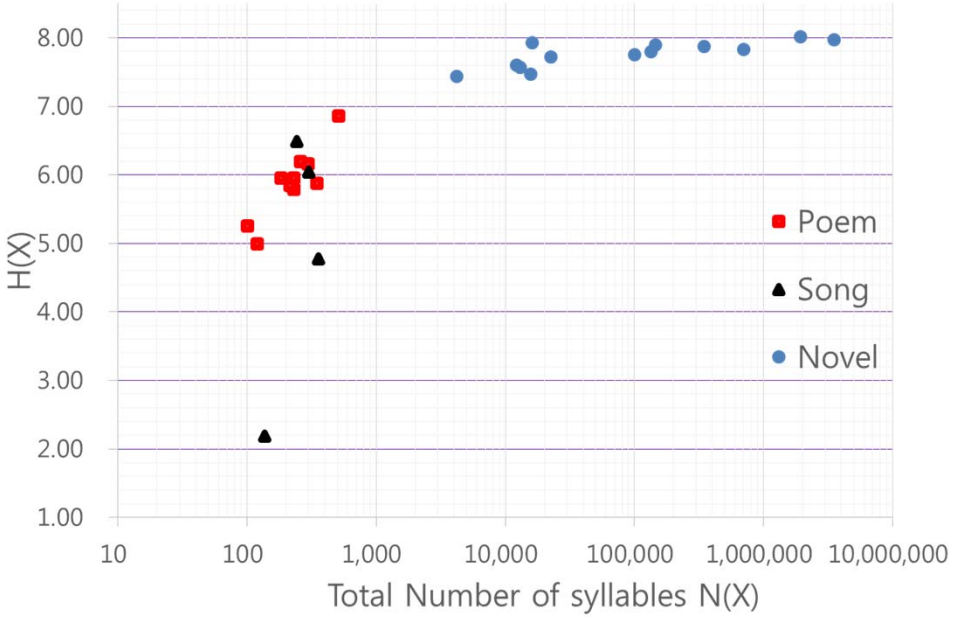
cocktail party problem

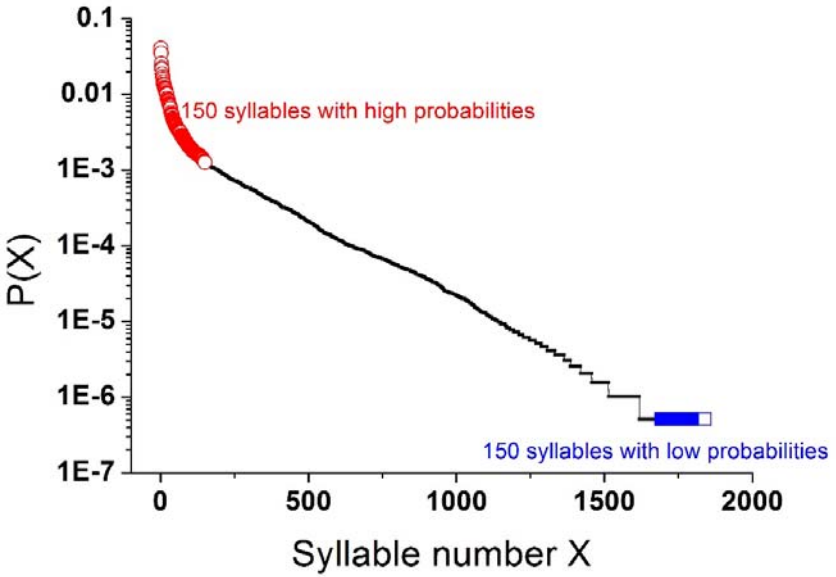
Automatic speech (complex sound) recognition is still far from human speech recognition

What is the effect of noise on the hearing of complex sounds such as syllables?



Syllable selection for hearing test





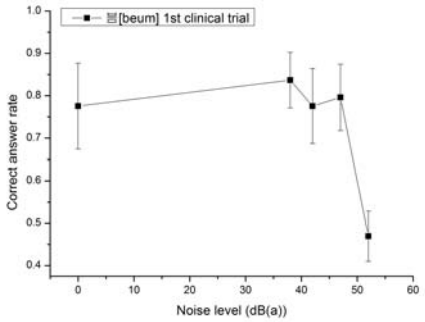
$$H(x) = -\sum_{x} p(x)\, log\big(p(x)\big)$$

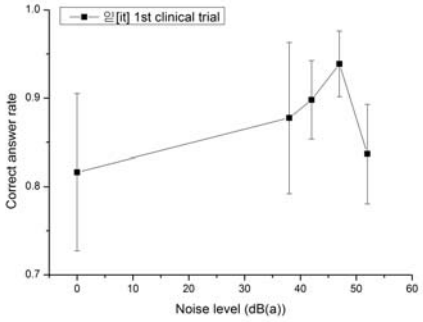
150 syllables with high probabilities and **150 syllables with low probabilities**

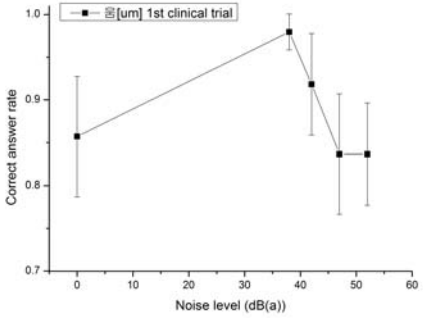
Usual syllable **Unusual syllable**

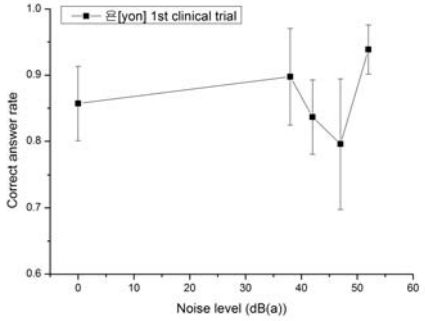
1st Clinical trial

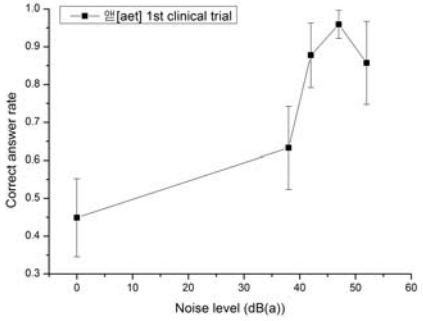
- **23 men and 26 women**
- **Male age range is 20~26, female age range is 19~24**
- **300 Korean syllables. (60 dB(a))**
- **5 white noise levels (38 dB(a), 42 dB(a), 47 dB(a), 52 dB(a), 57 dB(a))**





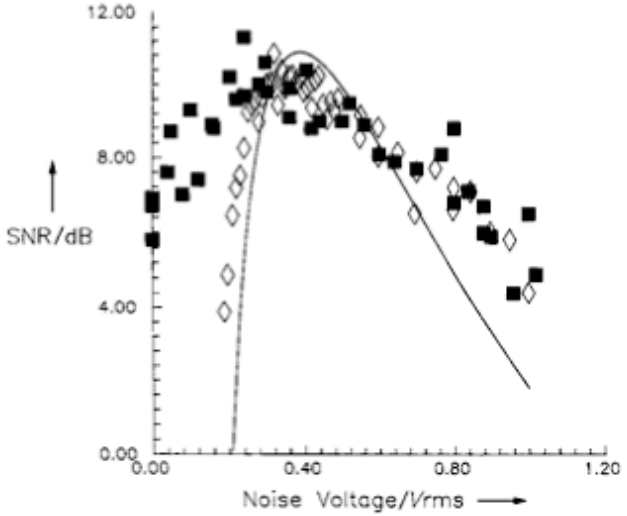






The behavior looks like stochastic resonance, but

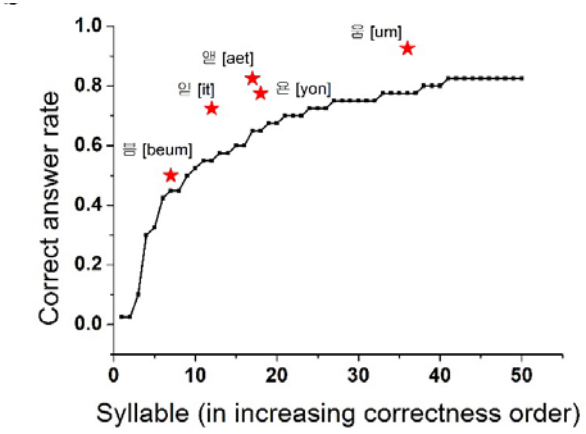
Stochastic resonance



Hanggi (2002) *Chem Phys Chem* 3: 285

In clinical trial, noise-enhanced syllable hearing appeared only for the specific syllables.

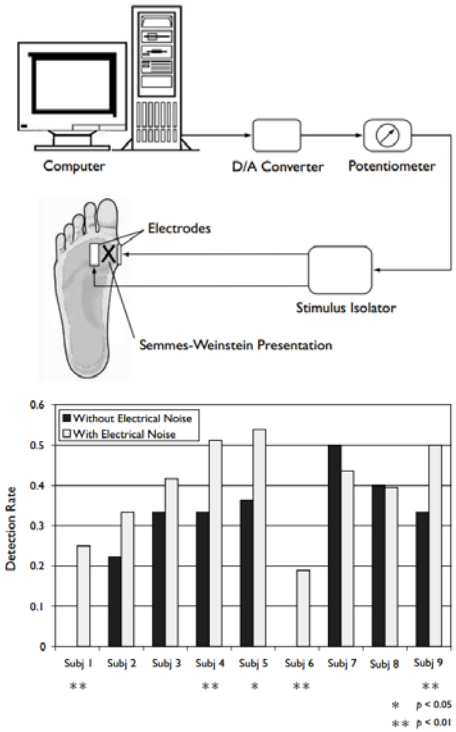
Noise-enhanced hearing syllables are rarely used syllables



However, these are not the cause of this phenomenon

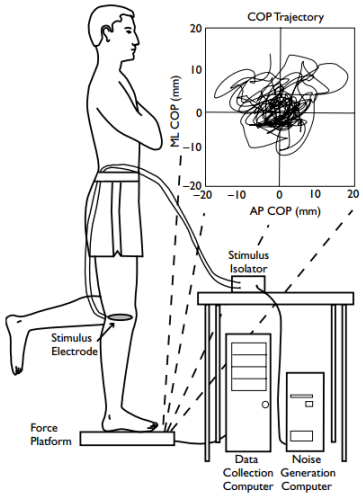
Noise can improve other cognitive abilities

Tactile sensation



Dhruv, Neel T., et al. "Enhancing tactile sensation in older adults with electrical noise stimulation." Neuroreport 13.5 (2002): 597-600.

Balance control



Sway measures	No stimulation	Stimulation	Improvement (%)
ML SD (mm)	4.6 ± 0.7	4.5 ± 0.7	3.8
ML max (mm)	13.0 ± 2.2	13.1 ± 2.6	−1.1
AP SD (mm)	6.3 ± 1.3	6.1 ± 1.4	3.7
AP max (mm)	20.0 ± 5.6	19.0 ± 5.3	5.4
Mean radius (mm)	6.9 ± 1.1	6.7 ± 1.2	3.3
Path length (mm/s)	37.1 ± 5.0	35.9 ± 5.8	3.1
Swept area (mm^2/s)	88.8 ± 25.5	81.9 ± 24.7	7.8

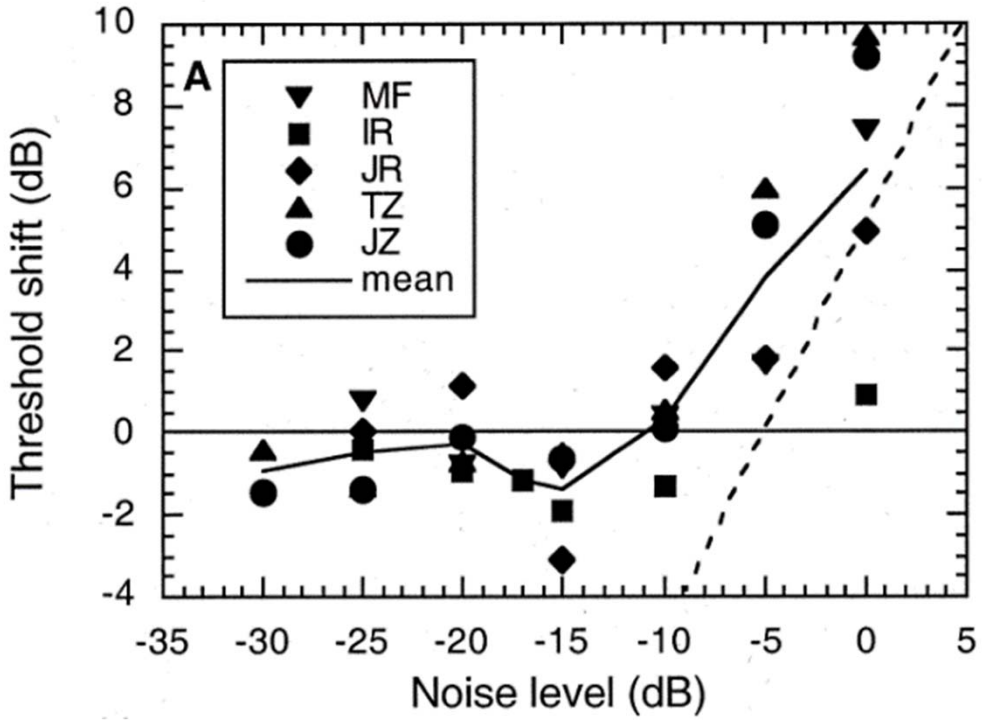
Gravelle, Denise C., et al. "Noise-enhanced balance control in older adults."Neuroreport 13.15 (2002): 1853-1856.

Vision

Moss, Frank, Lawrence M. Ward, and Walter G. Sannita. "Stochastic resonance and sensory information processing: a tutorial and review of application." Clinical neurophysiology 115.2 (2004): 267-281.

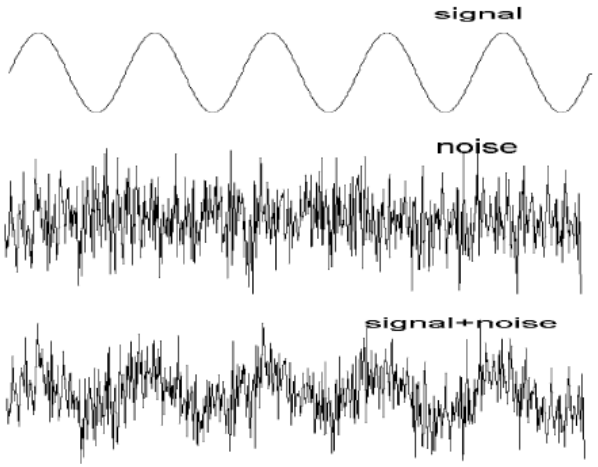
Addition of an appropriate amount of noise can actually improve cognitive abilities

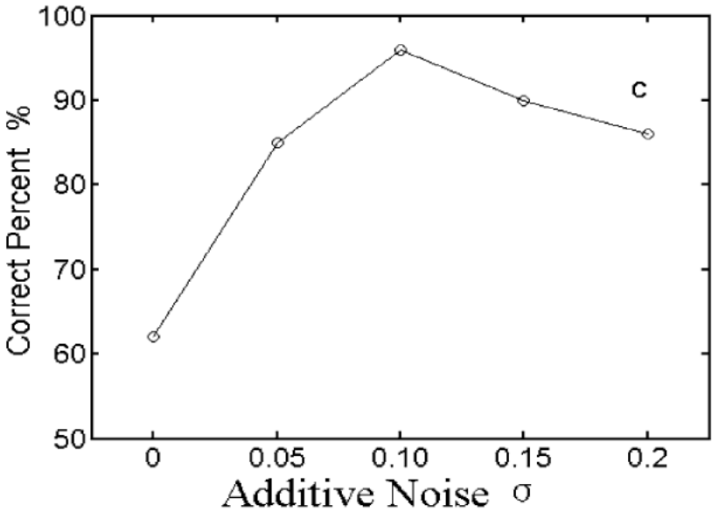
Appropriate amount of noise can help hearing pure tone sounds by lowering threshold



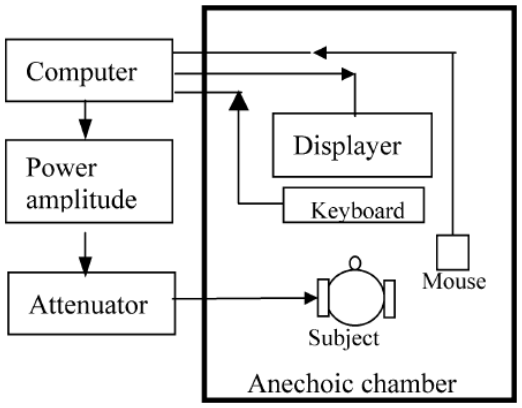
F. Zeng, Q. Fu, R. Morse, Brain Res. 869 (2000) 251.

Appropriate amount of noise can help hearing pure tone sounds by lowering threshold





Long, Zhang-Cai, et al. *Physics Letters A*323.5 (2004): 434-438.



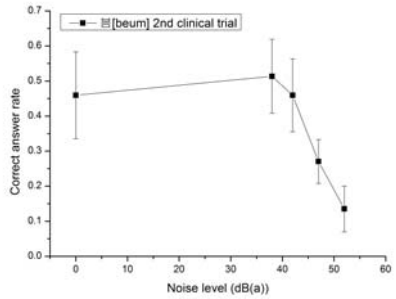
Correct percentages of detecting pure tones of different amplitudes with additive noise

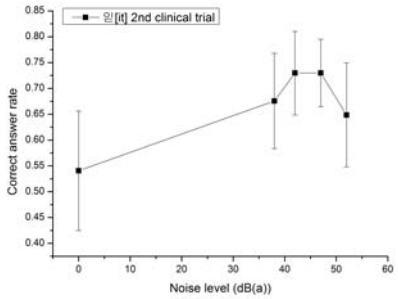
What happened?

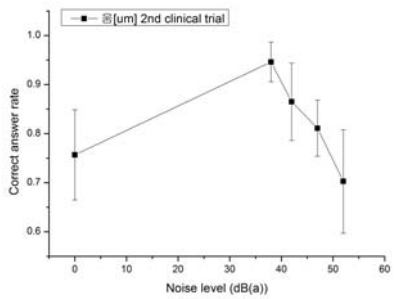
Response bias ?
Individual disorder ?
More attention with noise?

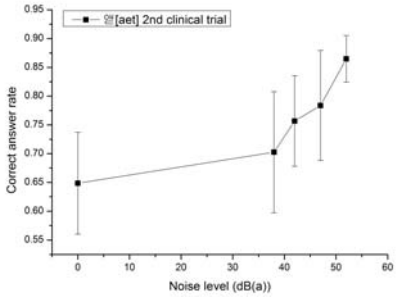
2nd Clinical trial

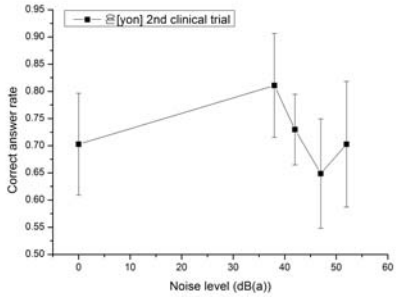
- **15 men and 22 women**
- **200 Korean syllables with 4 levels of white noise (syllable 201~300 from the 1st clinical trial and the 57 dB(a))**
- **The order of the syllable was different for each subject (Remove response bias)**



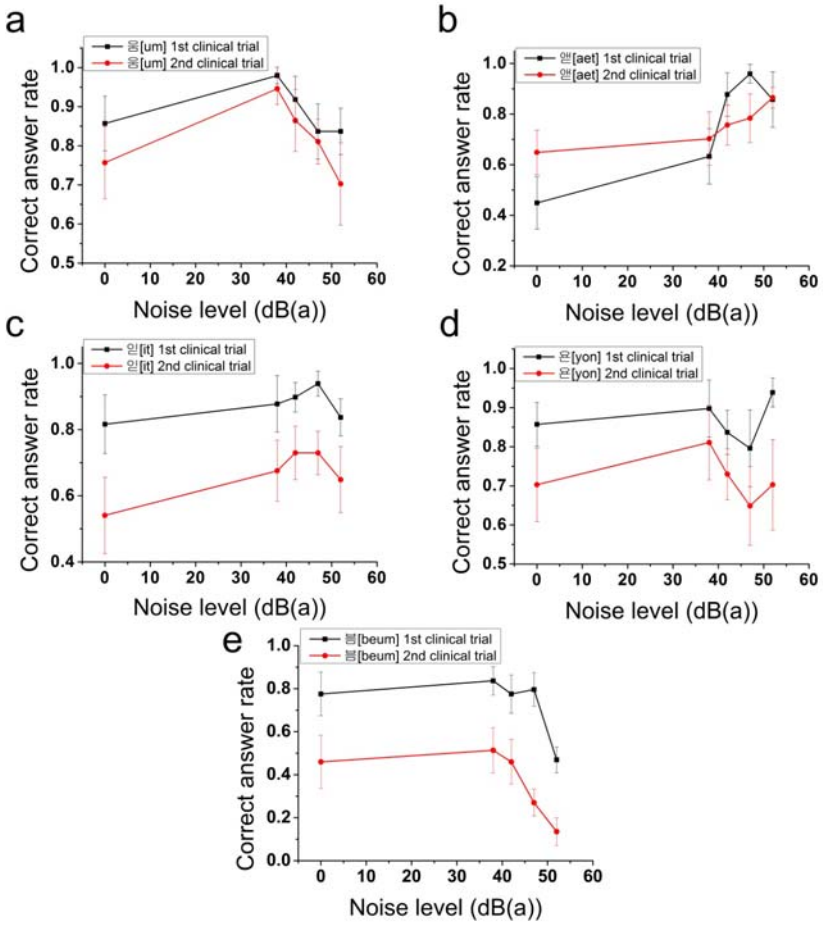








2nd Clinical trial



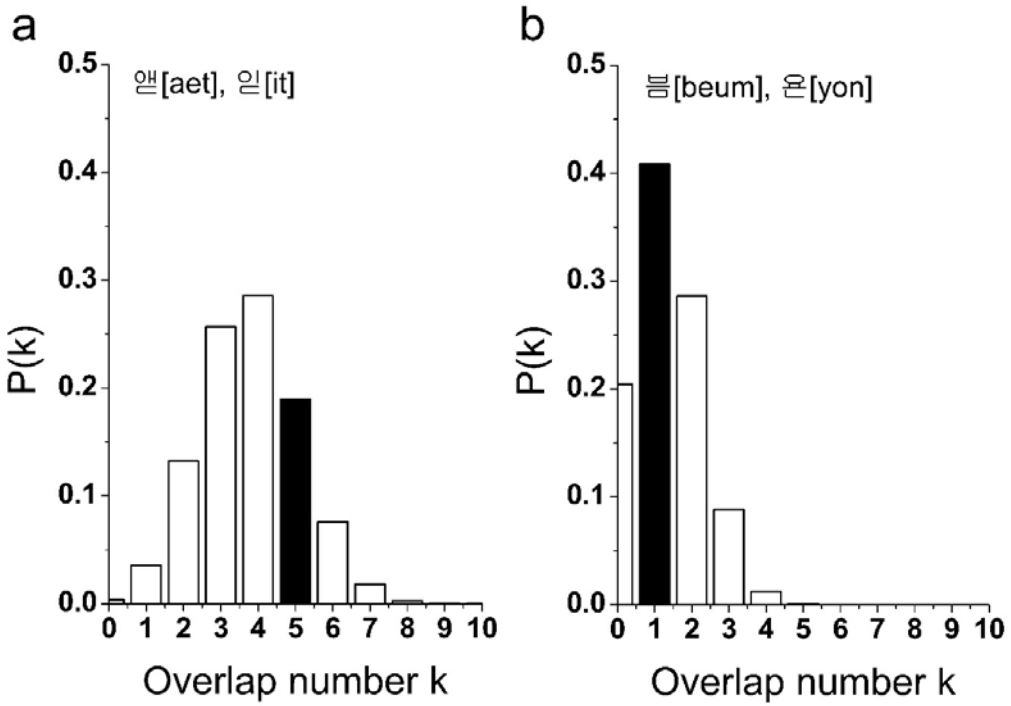
1st and 2nd clinical trial syllables have a similar increasing structure.

Syllable dependence or individual dependence

If the effect is due to individual disorders, there must b strong overlap between the people who have this effect for different syllables.

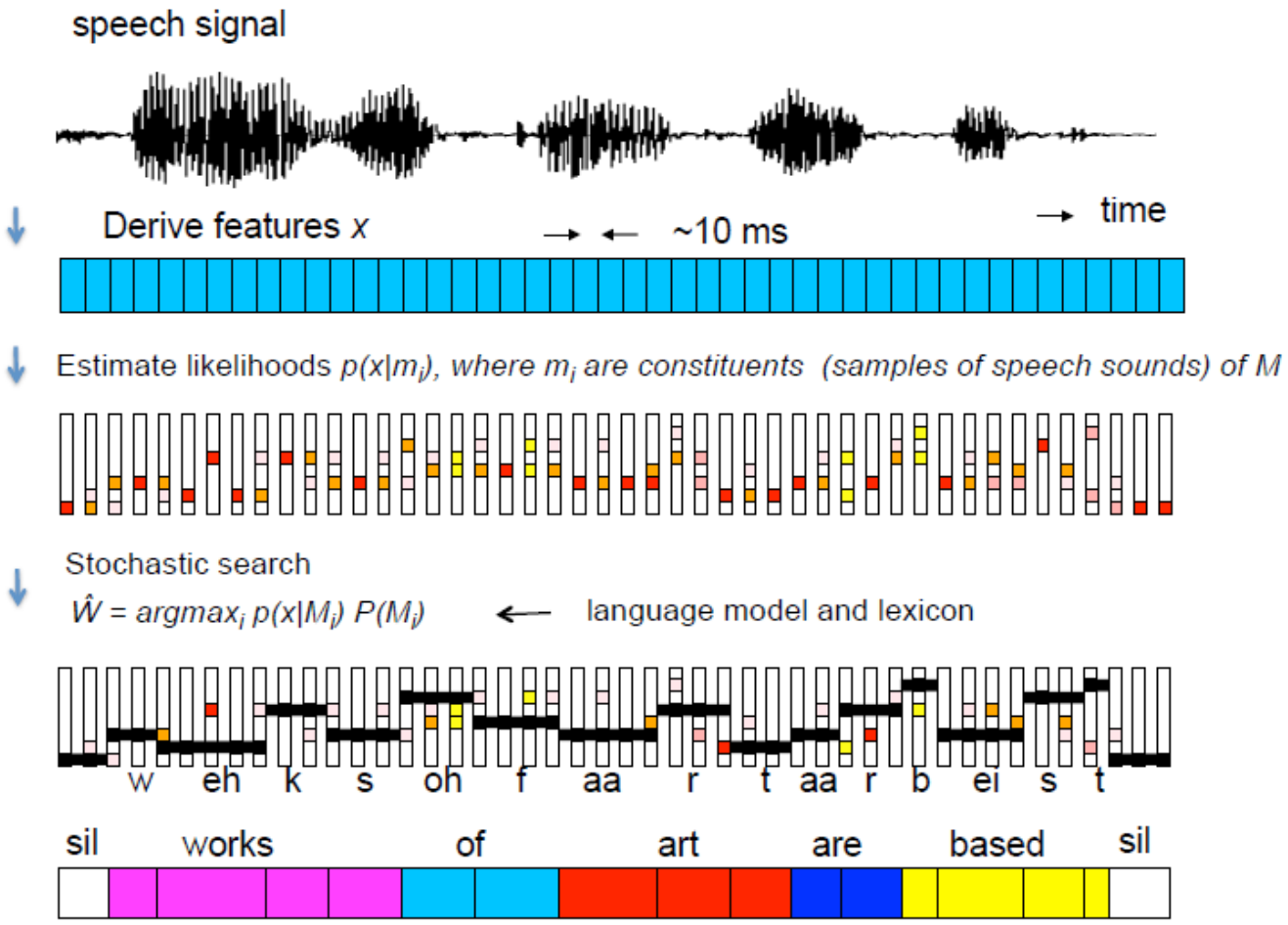
Number of overlap k between two randomly chosen groups of number of people m and n among a total N

$$P_k = \frac{\binom{m}{k}\binom{N-m}{n}}{\binom{N}{n}}$$



This overlap number is small and it can happen by chance

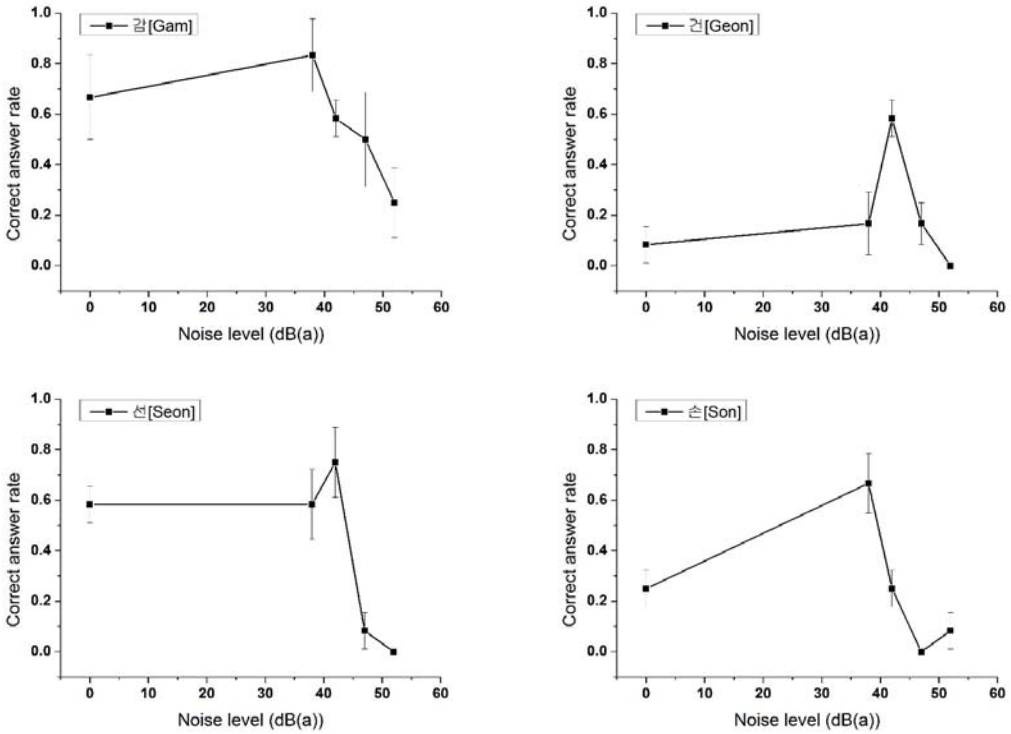
The application of the machine learning to speech recognition



Syllable recognition test in automatic speech recognition

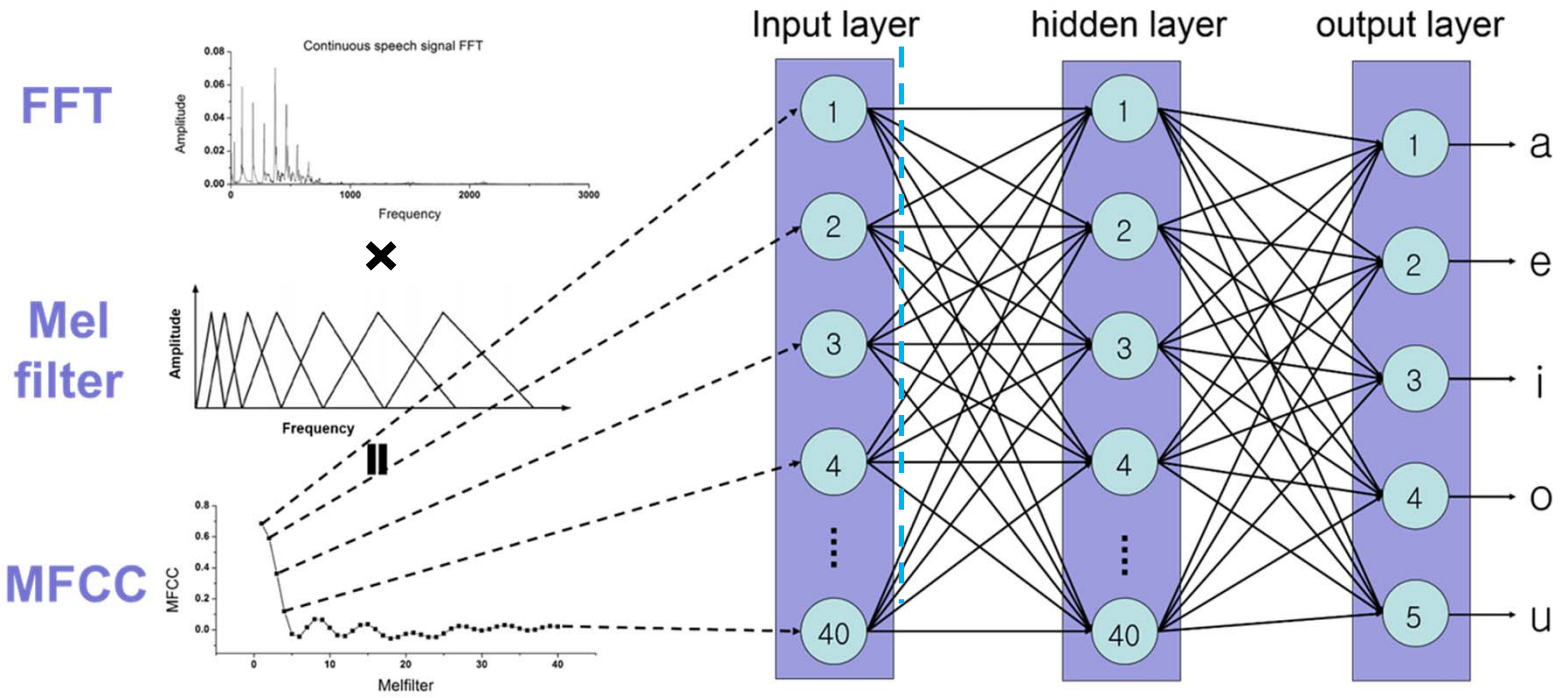
If noise-induced hearing enhancement of syllables comes from the **central nervous system**, it is natural to expect this phenomenon arise also in an **artificial speech recognition systems**.

- 200 Korean syllables (201~300 in 1st clinical trial test syllable were excluded).
- Test was repeated 17 times



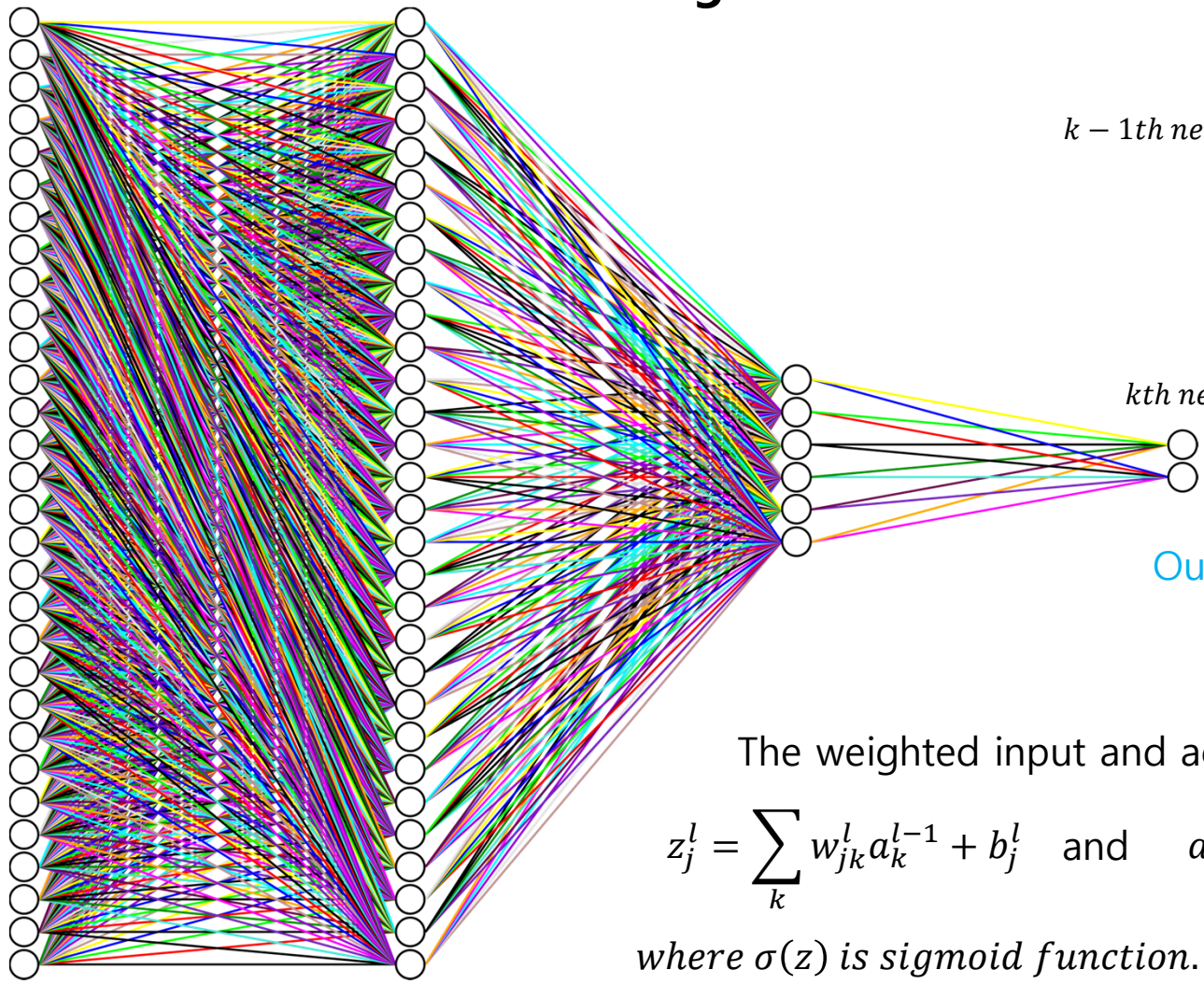
The noise-induced hearing enhancement also found in TTS-STT system

Syllable recognition test in artificial speech recognition



Can we see the same phenomenon in feedforward neural network?

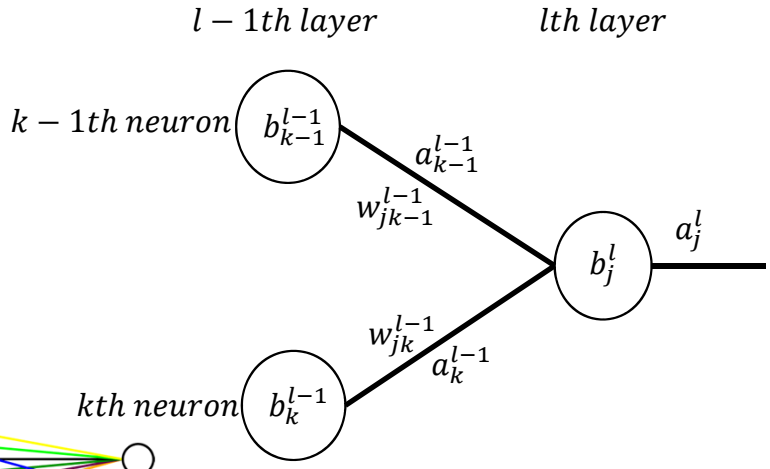
Machine Learning Architecture



Input layer

Hidden layer

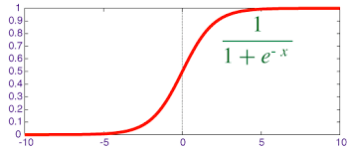
Output layer



The weighted input and activation

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l \quad \text{and} \quad a_j^l = \sigma(z_j^l)$$

$where\ \sigma(z)\ is\ sigmoid\ function.$

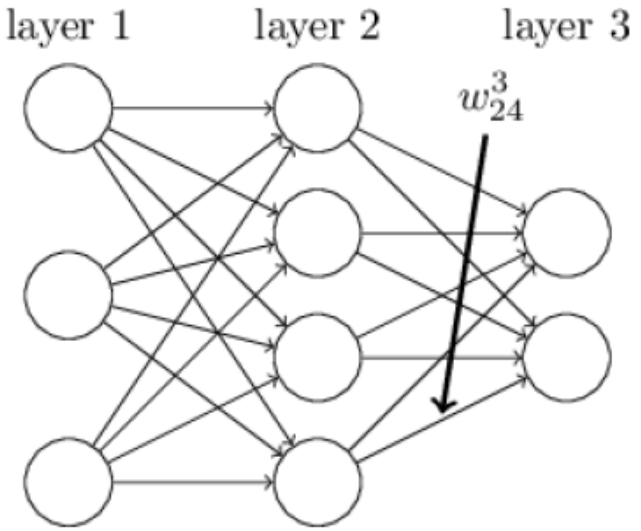


Training : Minimizing an object function (error)

$$C(w,b) \equiv \sum_x \|y(x) - a\|^2$$

Desired output

Calculated output



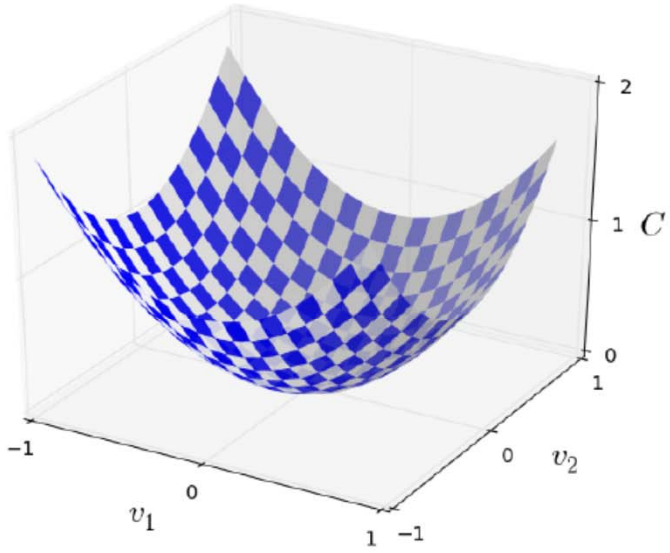
$$a^l_j = \sigma\left(\sum_k w^l_{jk} a^{l-1}_k + b^l_j\right)$$

activation

Weight matrix

bias

Gradient descent method



$$\Delta C \approx \nabla C \cdot \Delta v.$$

$$\Delta v = (\Delta v_1, \ldots, \Delta v_m)^T$$

$$\nabla C \equiv \left(\frac{\partial C}{\partial v_1}, \ldots, \frac{\partial C}{\partial v_m} \right)^T$$

$$\Delta v = -\eta \nabla C.$$

Learning : Repeated application of

$$w_k \rightarrow w'_k = w_k - \eta \frac{\partial C}{\partial w_k}$$

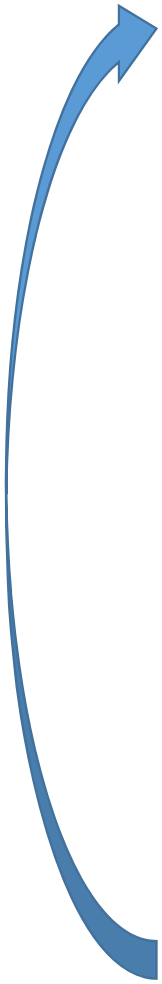
$$b_l \rightarrow b'_l = b_l - \eta \frac{\partial C}{\partial b_l}.$$

Back propgation

The equations of backpropagation

1. $\delta^L = \nabla_a C \odot \sigma'(z^L)$
2. $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$
3. $\frac{\partial C}{\partial b_j^l} = \delta_j^l$
4. $\frac{\partial \mathrm{C}}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l$

Hardamard product



1. **Input x:** Set the corresponding activation a^1 for the input layer.

2. **Feedforward:** For each $l = 2, 3, \ldots, L$ compute $z^l = w^l a^{l-1} + b^l$ and $a^l = \sigma(z^l)$.

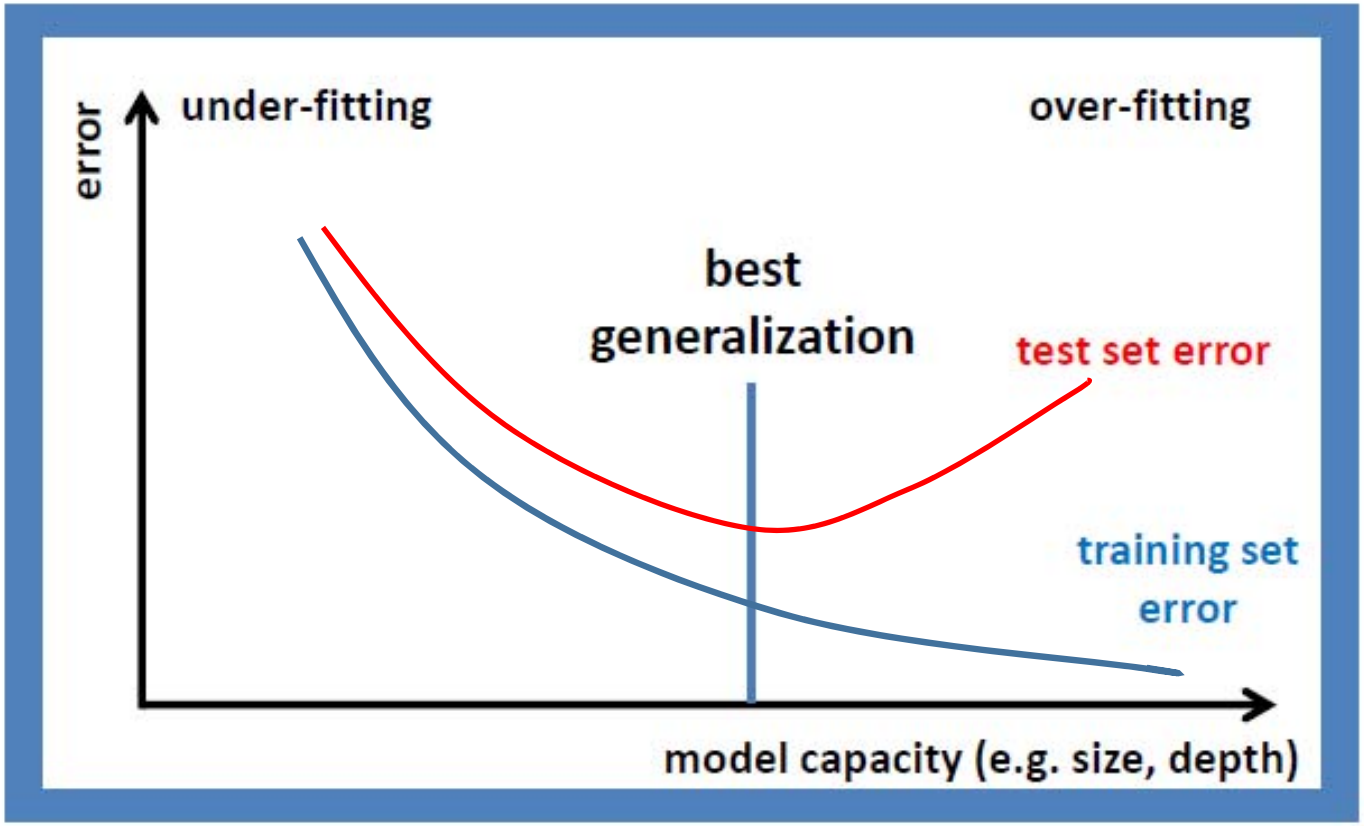
3. **Output error δ^L:** Compute the vector $\delta^L = \nabla_a C \odot \sigma'(z^L)$.

4. **Backpropagate the error:** For each $l = L-1, L-2, \ldots, 2$ compute $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$.

5. **Output:** The gradient of the cost function is given by $\frac{\partial C}{\partial w^l_{jk}} = a^{l-1}_k \delta^l_j$ and $\frac{\partial C}{\partial b^l_j} = \delta^l_j$.

6. **Gradient descent:** For each $l = L, L-1, \ldots, 2$ update the weights according to the rule $w^l \to w^l - \frac{\eta}{m} \sum_x \delta^{x,l} (a^{x,l-1})^T$, and the biases according to the rule $b^l \to b^l - \frac{\eta}{m} \sum_x \delta^{x,l}$.

Undertraining vs Overtraining



Most people in this field know this phenomenon but there is no mathematical proof.

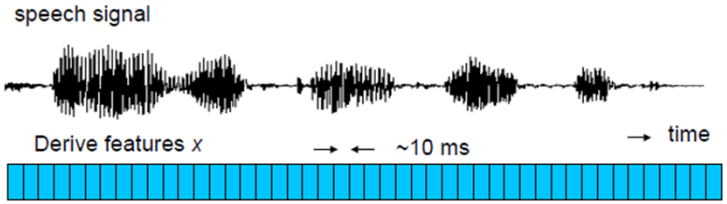
MFCC: Mel scale and Mel Frequency Cepstral Coefficient

- Mel scale

Humans are much better at discerning small changes in pitch at low frequencies than they are at high frequencies.

$$m(f) = 1125 \ln\left(1 + \frac{f}{700}\right) = 2595 log_{10}\left(1 + \frac{f}{700}\right)$$

- MFCC



1. Take the Discrete Fourier Transform of each frame.

$$S_i(f) = \sum_{n=1}^{N} s_i(n) h(n) e^{-\frac{i2\pi f n}{N}} \quad (1 \le f \le F)$$

where $S_i(n)$is the ith time − domain frame and $h(n)$ is Hamming window function.

2. Multiply each filterbank with the power spectrum and then add up the coefficients.

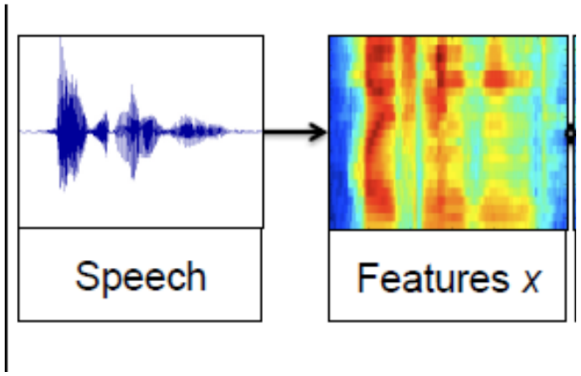
$$\tilde{S}_{ij} = \sum_{f=0}^{N/2} S_i(f) M_j(f)$$

where $M_j(f)$ is the jth mel filterbank.

3. Take the log of each $\tilde{S}_{ij}$ and take the Discrete Cosine Transform.

$$c_i(n) = \frac{1}{\sqrt{J}} \sum_{j=0}^{J} \log(\tilde{S}_{ij}) \cos(\frac{(2j+1)n}{2J})$$

where J is the number of mel filterbanks.

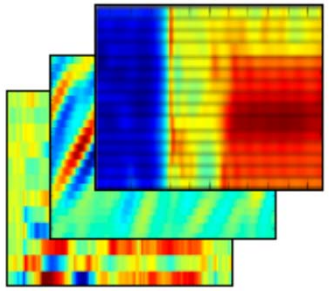


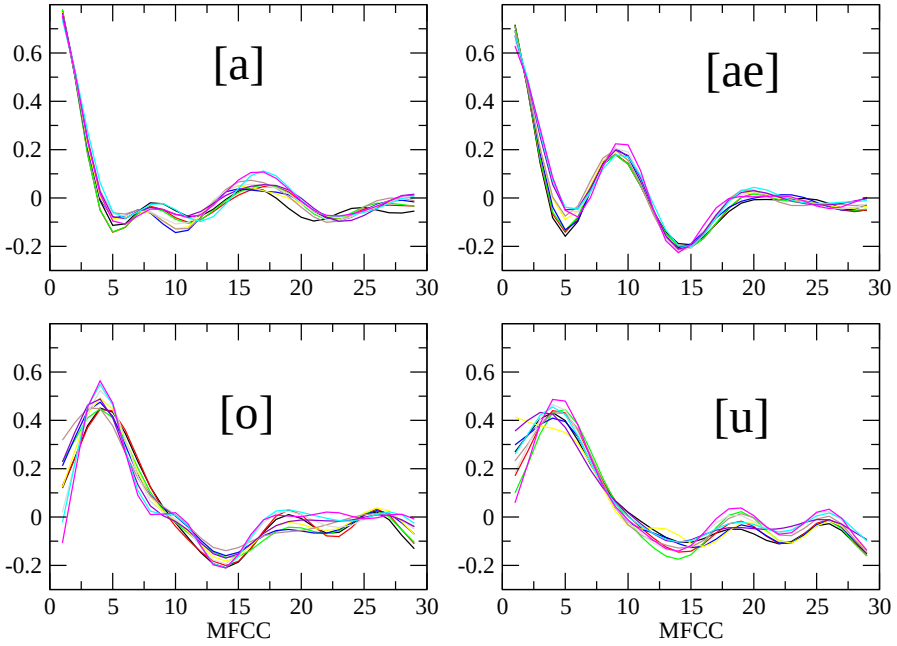
MFCC

The pattern for different phonemes are more *visible* in MFCC.

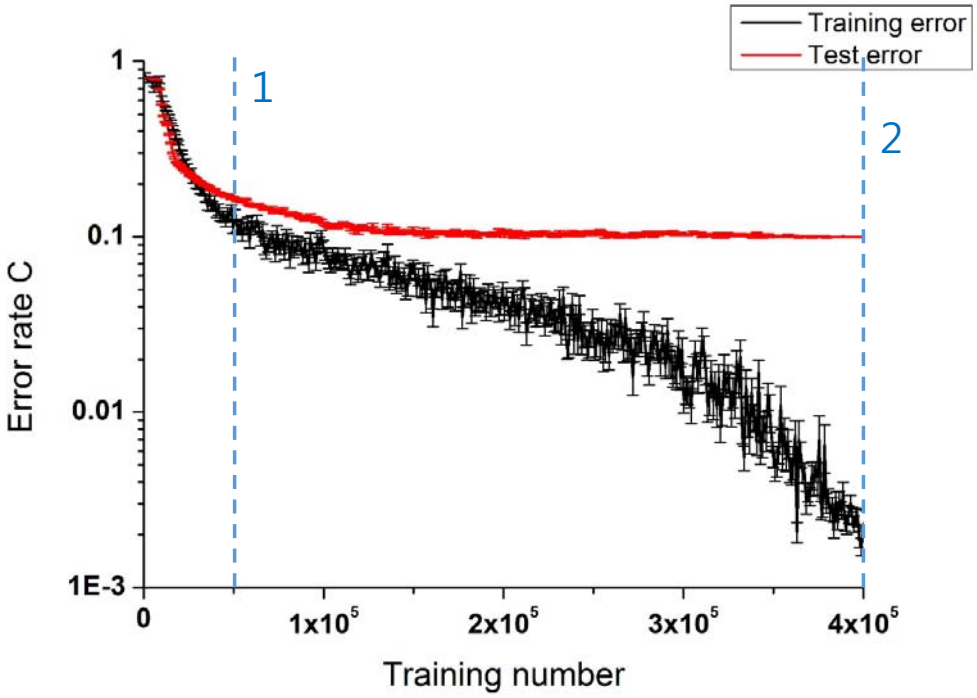
Bioinspired: Cochlea

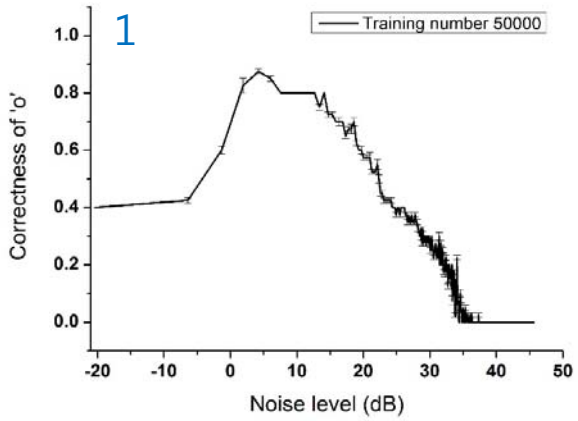
Much simpler than spectrum

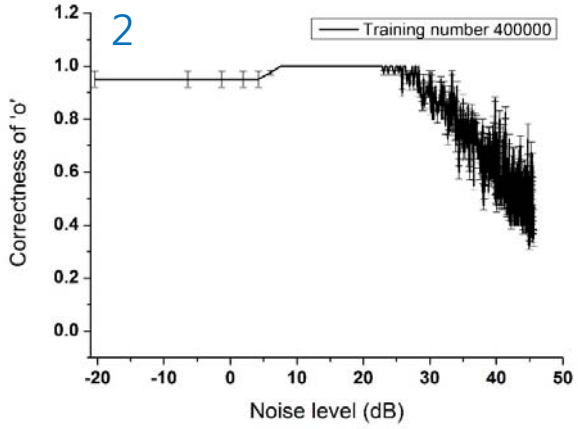




Noise-induced hearing enhancement appears in insufficiently trained network







Syllable *o* have a maximum correctness at a specific noise intensity in the feedforward neural network.

Mathematical model made not that easy

$$S_i(f) = \sum_{n=1}^{N} s_i(n)h(n)e^{-\frac{i2\pi fn}{N}} \quad (1 \le f \le F)$$

$$\tilde{S}_{ij} = \sum_{f=0}^{N/2} S_i(f)M_j(f)$$

$$c_i(n) = \frac{1}{\sqrt{J}} \sum_{j=0}^{J} \log(\tilde{S}_{ij}) \cos(\frac{(2j+1)n}{2J})$$

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l \qquad a_j^l = \sigma(z_j^l)$$

$$C(w,b) \equiv \frac{1}{2n} \sum_x \|y(x) - a\|^2$$

$$w_k \rightarrow w_k' = w_k - \eta \frac{\partial C}{\partial w_k}$$

$$b_l \rightarrow b_l' = b_l - \eta \frac{\partial C}{\partial b_l}.$$