# Coffee Time

May 12, 2017

Xiaolin Hu
xlhu@tsinghua.edu.cn

# AI beats human in Go

nature
At last — a computer program that can beat a champion Go player
ALL SYSTEMS GO

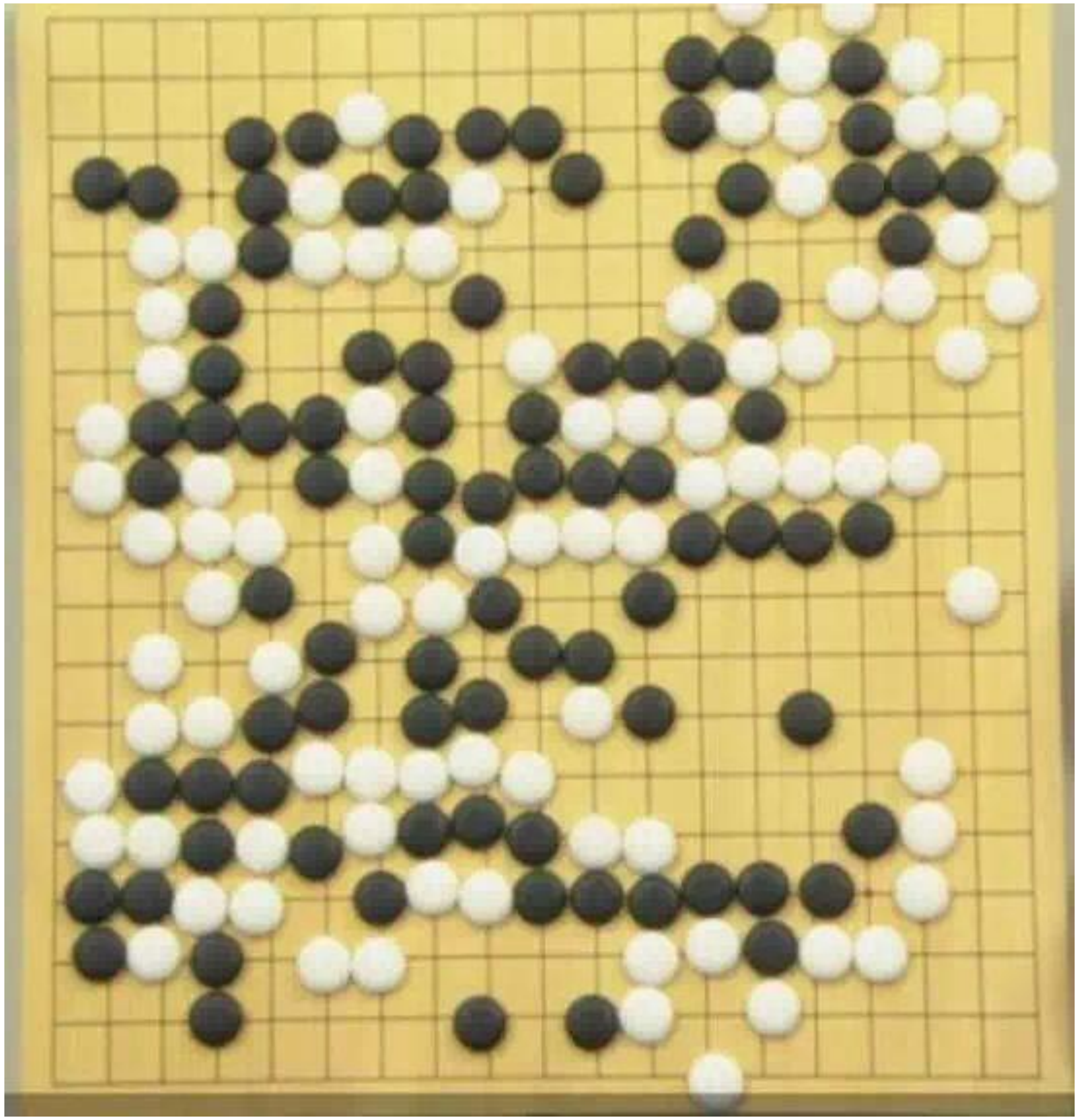
28 January 2016

- AlphaGo赢欧洲围棋冠军樊麾(2015年10月5号 )
- 4:1赢世界围棋冠军李世石
- 2016年末2017年初，在中国棋类网站上以Master为注册帐号与中日韩数十位围棋高手进行快棋对决，连续60局无一败绩。

# What does it mean?

- 1928年，来自中国的天才少年**吴清源**登陆日本，虽然之前吴少年在中国已经战胜过多位日本低段棋手，但日本围棋的权威们依然将信将疑。直到吴清源横扫日本棋界，一路打到当时世界围棋第一人，本因坊秀哉的棋盘前。大家才认可天才少年吴清源的超凡实力，认可他对于当时围棋理论的突破。吴少年从速度和效率的思考方式解构了日本围棋四百年来，以小目定式为基础的围棋理论体系。
- **AlphaGo这个“机器人异类“，和80年前日本棋坛看少年吴清源这个“支那人异类”是类似的。**先是不相信，后来是担心，最后变成佩服。AlphaGo初出世，不是跟人类职业高手学的棋，下出来的手段自然不是职业高手熟悉的路数，他有些棋一出来就技惊四座，有些棋走出来憨态可鞠，有些棋也是些“小臭棋”。。。你使出手段试他几下，他有时胸有成竹兵来将挡，有时摇摇摆摆但应对无误，你也占不到什么大便宜，走着走着，忽然发现他的局势领先了。。。原来AlphaGo有他自己下棋的逻辑和方法，和我们现在的理论不太一样，其实非常厉害。这一幕，穿越回80年前和吴少年对局的日本棋手的感觉，可能是非常像的。

**摘自《少年吴清源之AlphaGo重生》新浪体育 褚达晨**



AlphaGo
执黑

# What does it mean?

- 人工智能的一大飞跃
- 意义比1997年IBM的深蓝赢世界象棋冠军重大
  – 不是胜在机器的运算速度，而是胜在了模拟人类感知和思考的功能
- 主要技术
  – Deep neural network
  – Reinforcement learning
  – Monte Carlo Tree Search

# Major components

- 走棋网络（Policy Network），给定当前局面，预测/采样下一步的走棋。
- 快速走子（Fast rollout），目标和1一样，但在适当牺牲走棋质量的条件下，速度要比1快1000倍。
- 估值网络（Value Network），给定当前局面，估计是白胜还是黑胜。
- 蒙特卡罗树搜索（Monte Carlo Tree Search，MCTS)，把以上这三个部分连起来，形成一个完整的系统。

By Facebook 田渊栋

## 人类智商的最后阵地失守！？

# Topic 12: Introduction to Deep Learning

胡晓林
Dept. of Computer Science and Technology
Tsinghua University

## Outline

- Background
- Multi-layer perceptron
- Convolutional neural nets
- Restricted Boltzmann machine
- Deep belief network
- Applications

Supervised learning

Unsupervised learning

# 深度学习广受欢迎

## 学术界

|  |  |  |
|---|---|---|
| Science | ICML | Toronto University |
| Nature | NIPS | Stanford University |
| IEEE T PAMI | CVPR | MIT |
| IEEE T MM | ICCV | Princeton |
| IEEE T IP | IJCAI | Harvard |
| IJCV | AAAI | New York University |
| JMLR | ACL | Montreal University |

Hinton

LeCun

Bengio

Ng

## 工业界

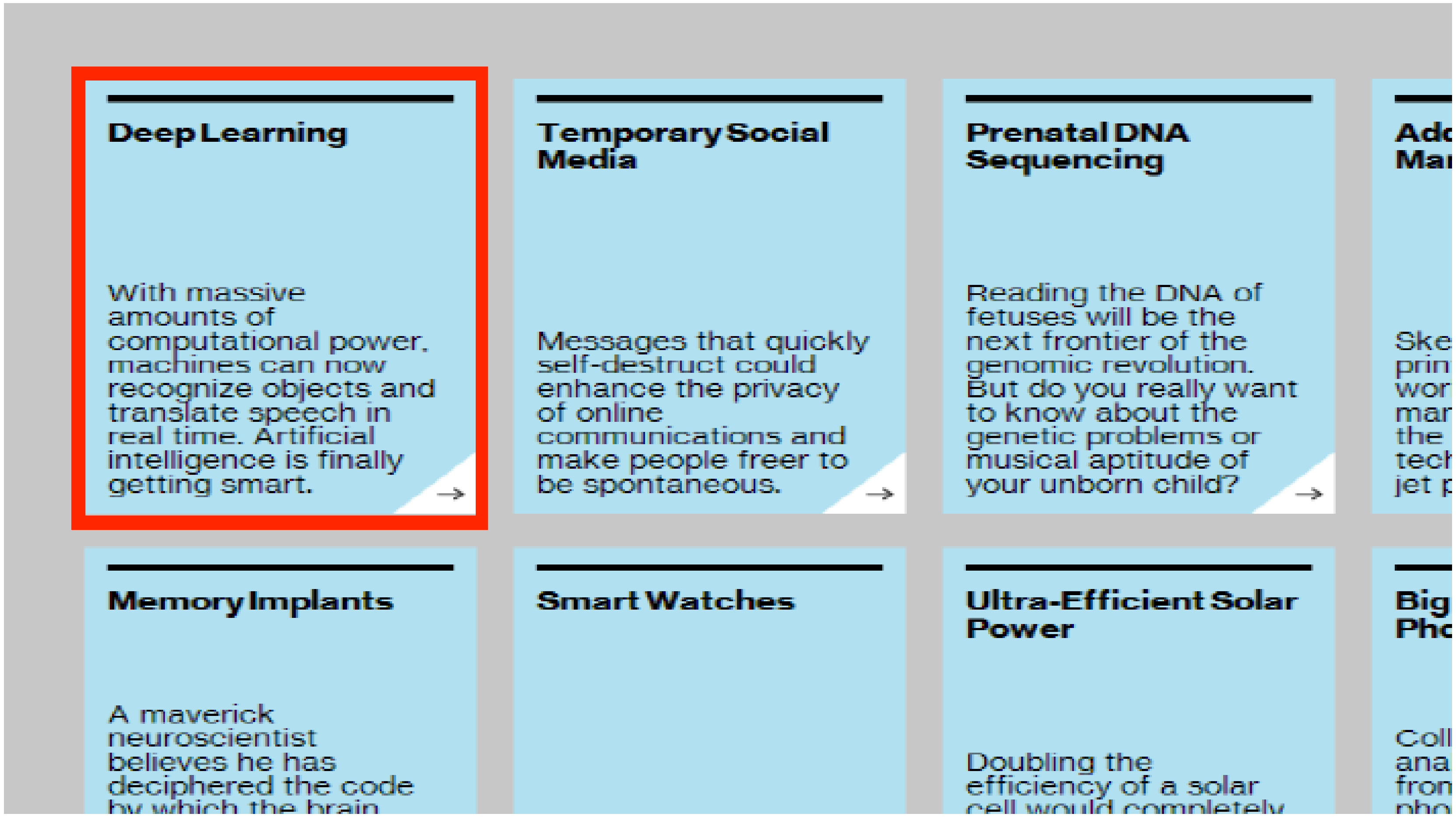
Google Facebook 阿里巴巴 Alibaba.com HUAWEI NVIDIA …

Microsoft Bai du 百度 Tencent 腾讯 AMD

MIT Technology Review

# 10 BREAKTHROUGH TECHNOLOGIES 2013

Intr



**Deep Learning**

With massive amounts of computational power, machines can now recognize objects and translate speech in real time. Artificial intelligence is finally getting smart. →

**Temporary Social Media**

Messages that quickly self-destruct could enhance the privacy of online communications and make people freer to be spontaneous. →

**Prenatal DNA Sequencing**

Reading the DNA of fetuses will be the next frontier of the genomic revolution. But do you really want to know about the genetic problems or musical aptitude of your unborn child? →

**Add Mar**

Ske prin wor mar the tech jet p

**Memory Implants**

A maverick neuroscientist believes he has deciphered the code by which the brain

**Smart Watches**

**Ultra-Efficient Solar Power**

Doubling the efficiency of a solar cell would completely

**Big Pho**

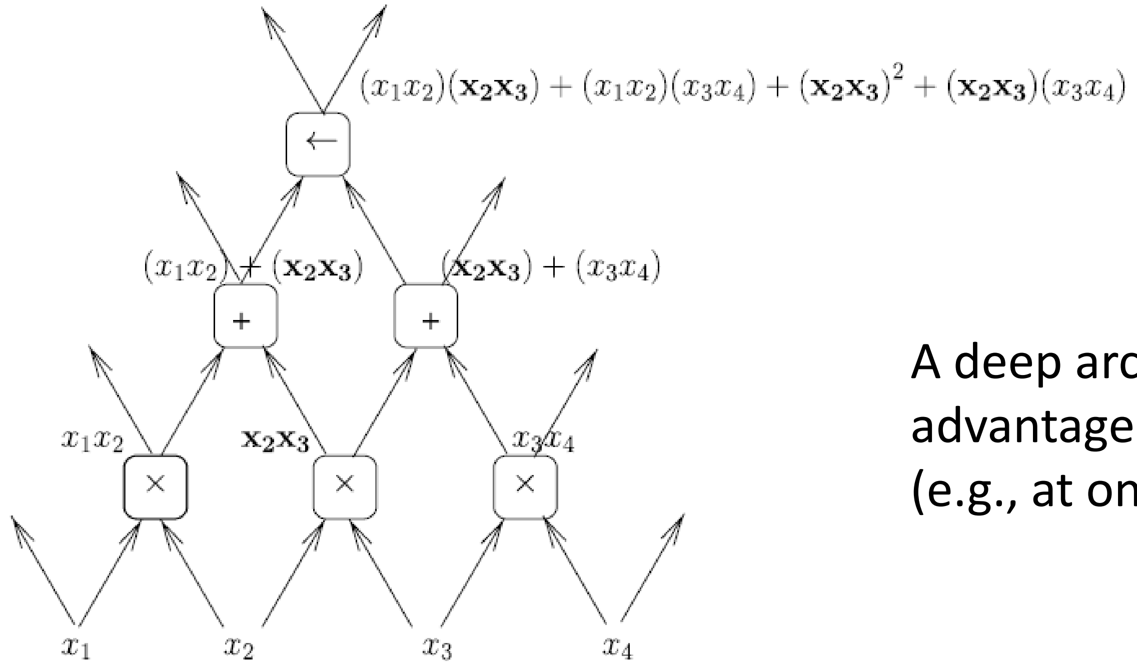
Coll ana fron pho

http://www.technologyreview.com/featuredstory/513696/deep-learning/

## Why go deep?

- Data are often high-dimensional.
- There is a huge amount of structure in the data, but the structure is too complicated to be represented by a simple model.
- Insufficient depth can require more computational elements than architectures whose depth matches the task.
- Deep nets provide simpler but more descriptive model of many problems.

# Computational complexity



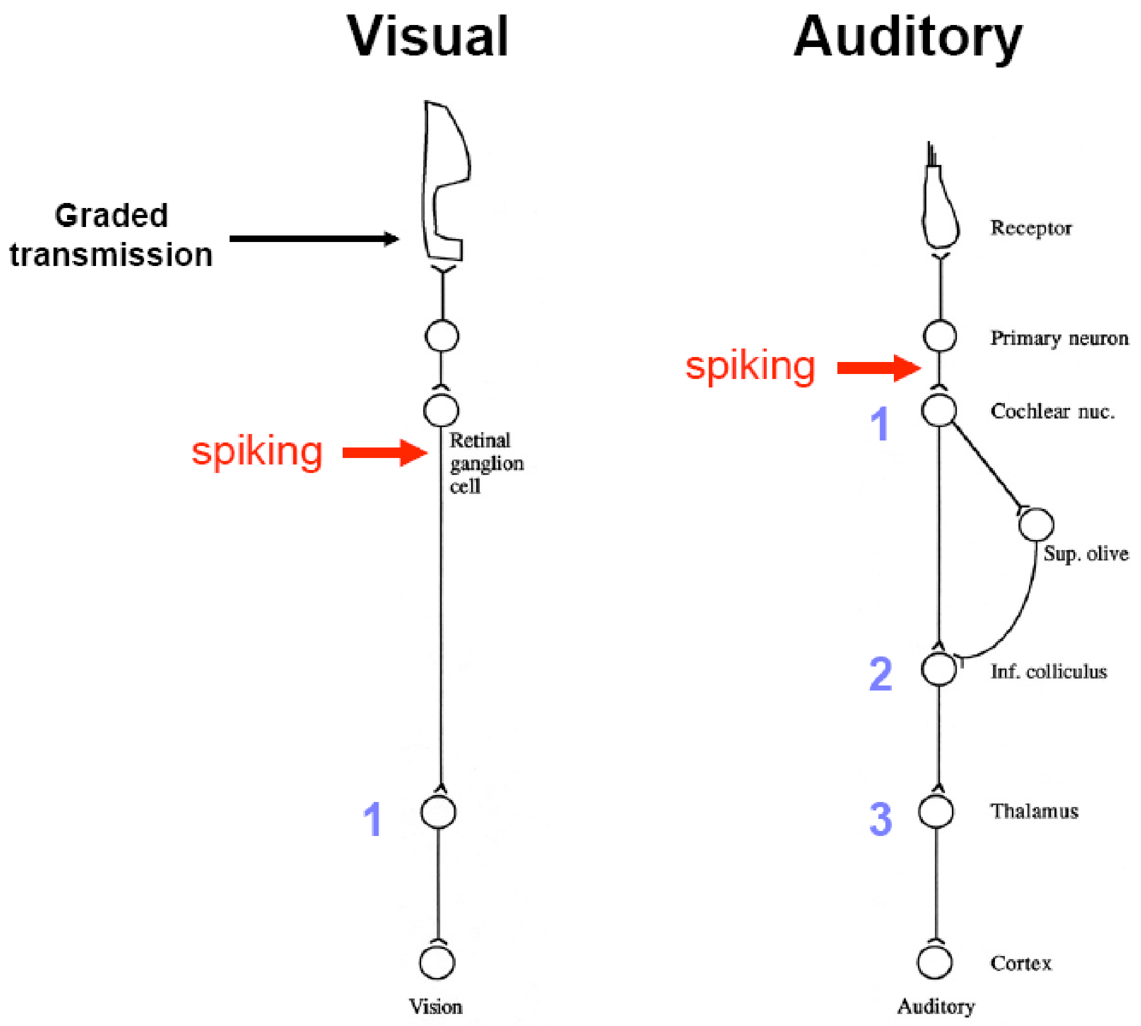
$(x_1x_2)(\mathbf{x_2x_3}) + (x_1x_2)(x_3x_4) + (\mathbf{x_2x_3})^2 + (\mathbf{x_2x_3})(x_3x_4)$
$\leftarrow$
$(x_1x_2) + (\mathbf{x_2x_3})$
$(\mathbf{x_2x_3}) + (x_3x_4)$
$+$
$+$
$x_1x_2$
$\mathbf{x_2x_3}$
$x_3x_4$
$\times$
$\times$
$\times$
$x_1$
$x_2$
$x_3$
$x_4$

A deep architectures can be advantageous if some computations (e.g., at one level) can be shared.

When a function can be compactly represented by a deep architecture, it might need a very large architecture to be represented by an insufficiently deep one.

Bengio, Learning Deep Architectures for AI, 2009

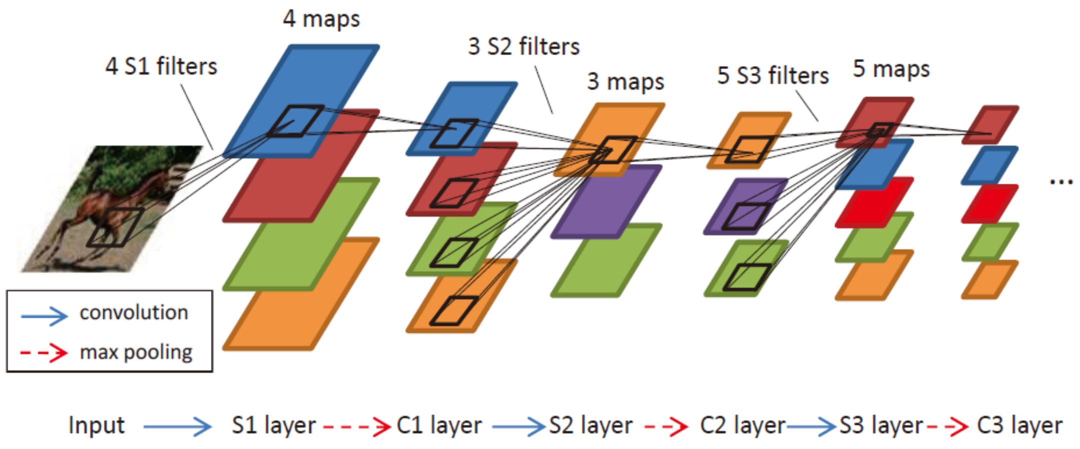
# Sensory information processing in human brain



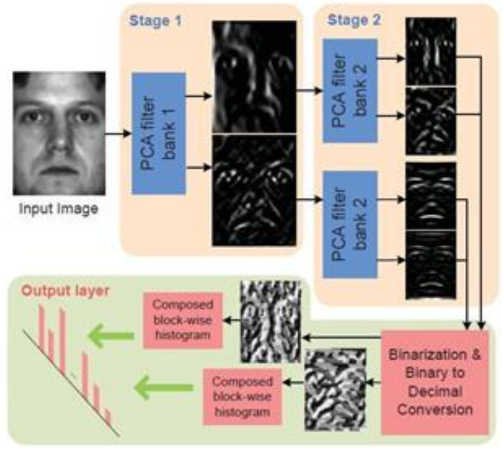
# Definition of deep learning

- 狭义
  - 人工神经网络
    - 前馈网络：多层感知机（MLP）、卷积神经网络（CNN）
    - 反馈网络：Elman网络、长短时记忆网络（LSTM）
- 广义
  - 层次化的机器学习模型

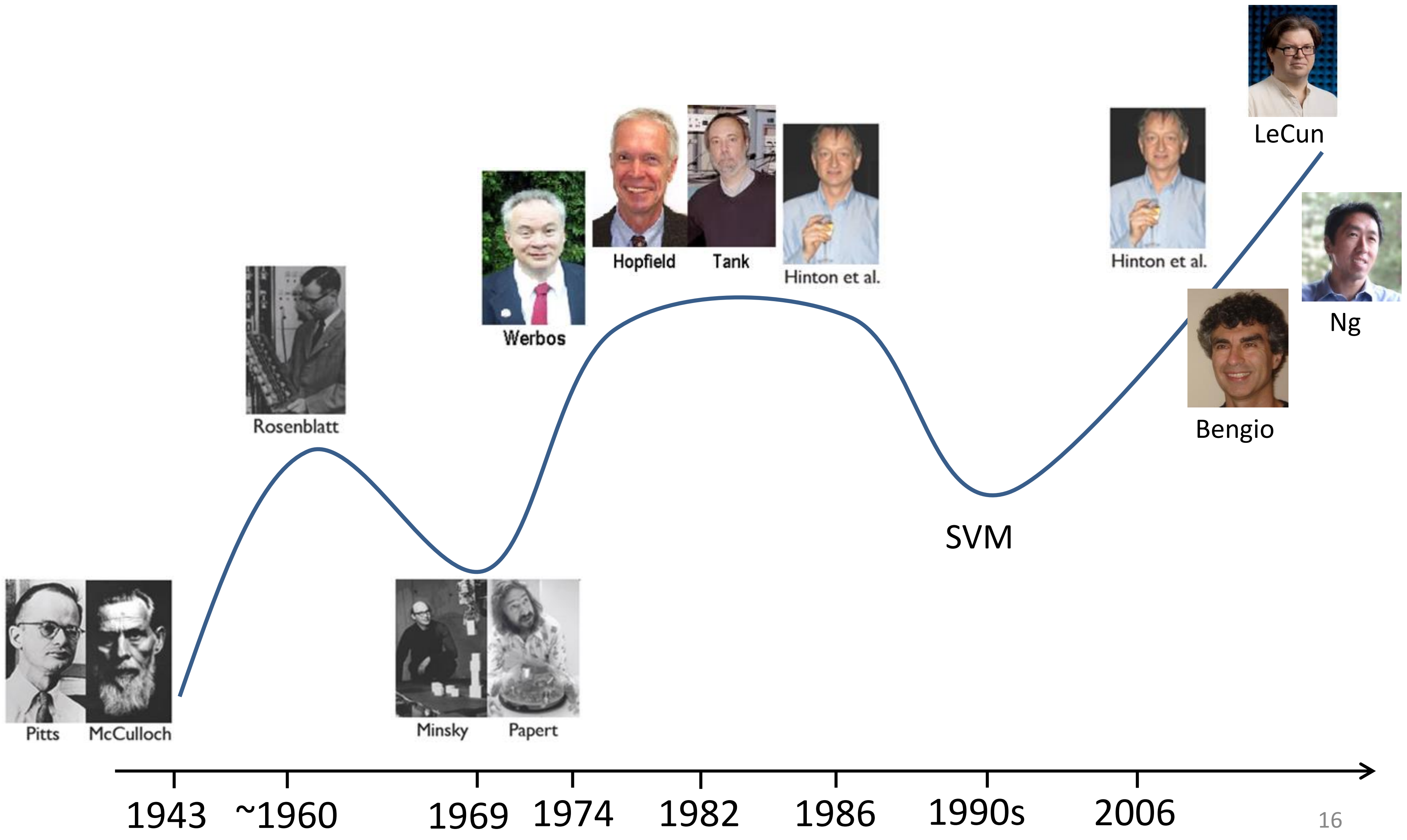
Sparse HMAX (Hu et al., 2014)



PCA net (Chan et al., 2014)


Stage 1
Stage 2
Input Image
PCA filter bank 1
PCA filter bank 2
PCA filter bank 2
Output layer
Composed block-wise histogram
Composed block-wise histogram
Binarization & Binary to Decimal Conversion

# history


Pitts
McCulloch
Rosenblatt
Minsky
Papert
Werbos
Hopfield
Tank
Hinton et al.
SVM
Hinton et al.
LeCun
Ng
Bengio
1943
~1960
1969
1974
1982
1986
1990s
2006

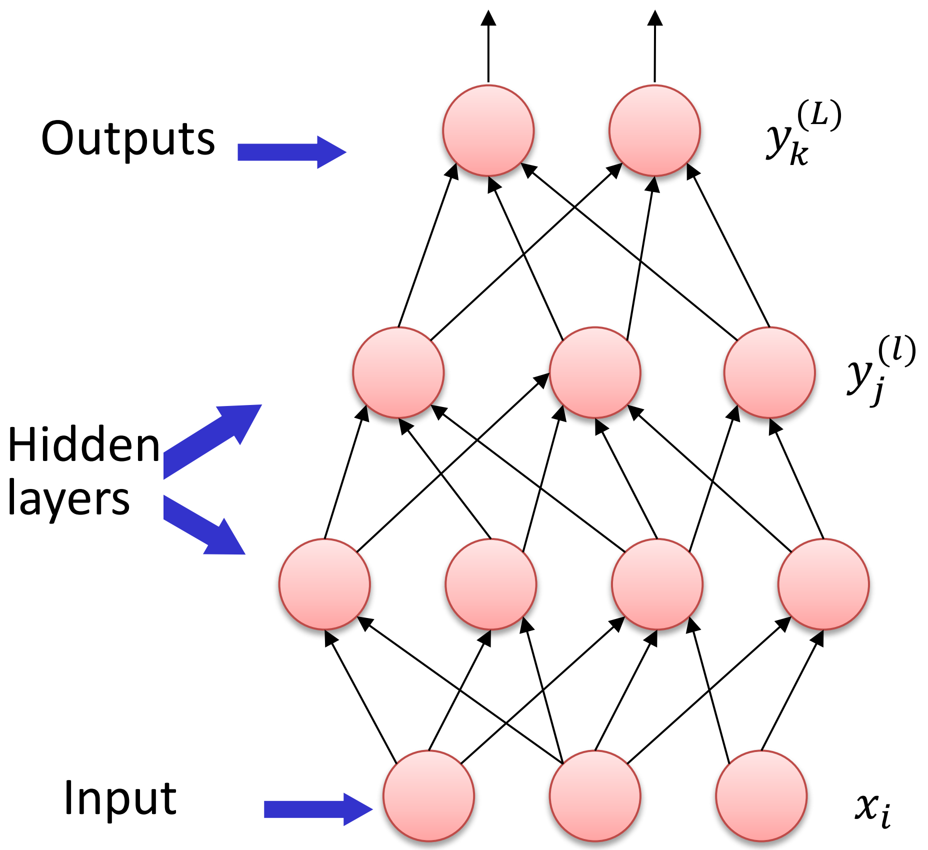
## Outline

- Background
- Multi-layer perceptron
- Convolutional neural nets
- Restricted Boltzmann machine
- Deep belief network
- Applications

$\left.\begin{array}{l}\text{Multi-layer perceptron}\\ \text{Convolutional neural nets}\end{array}\right\}$ Supervised learning

$\left.\begin{array}{l}\text{Restricted Boltzmann machine}\\ \text{Deep belief network}\end{array}\right\}$ Unsupervised learning

# Multi-layer Perceptron (MLP)



- There are a total of $L$ layers except the input
- Connections:
    - Fully connections between layers
    - No feedback connections between layers
    - No lateral connections in the same layer
- Every neuron receives input from previous layer and fire according to an activation function

# Activation functions

- Logistic function

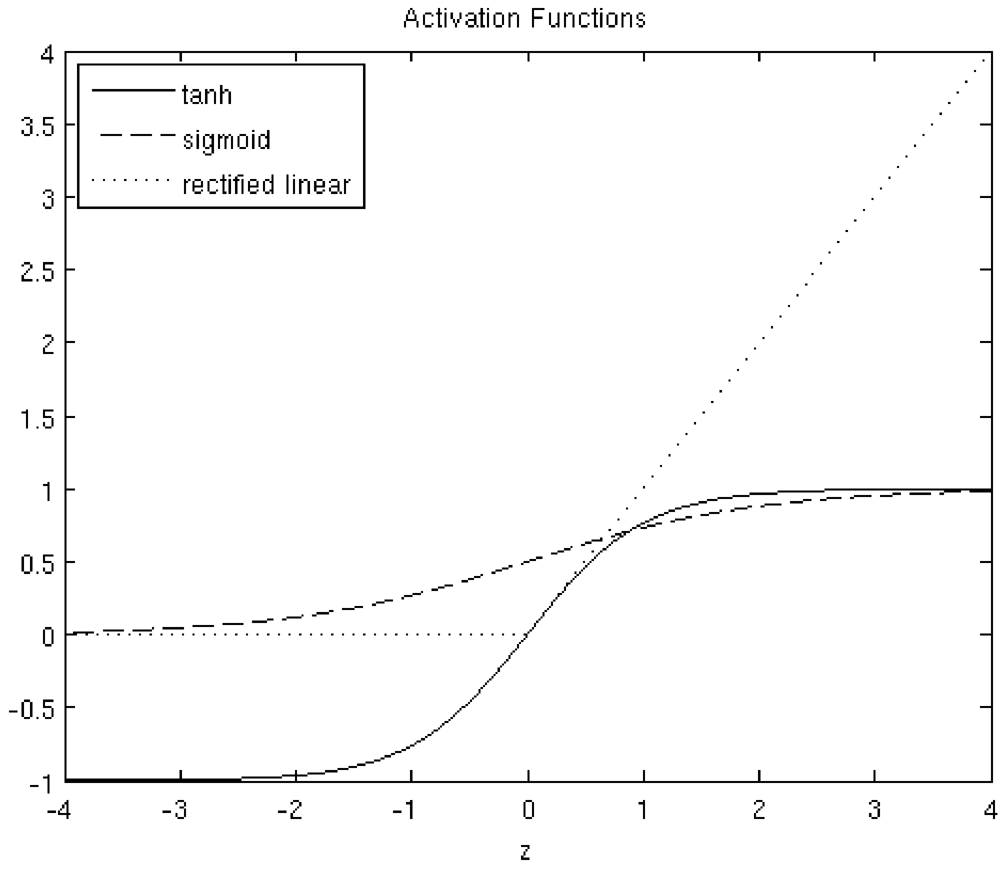
$$f(z) = \frac{1}{1+\exp(-z)}$$

- Hyperbolic tangent, or tanh, function

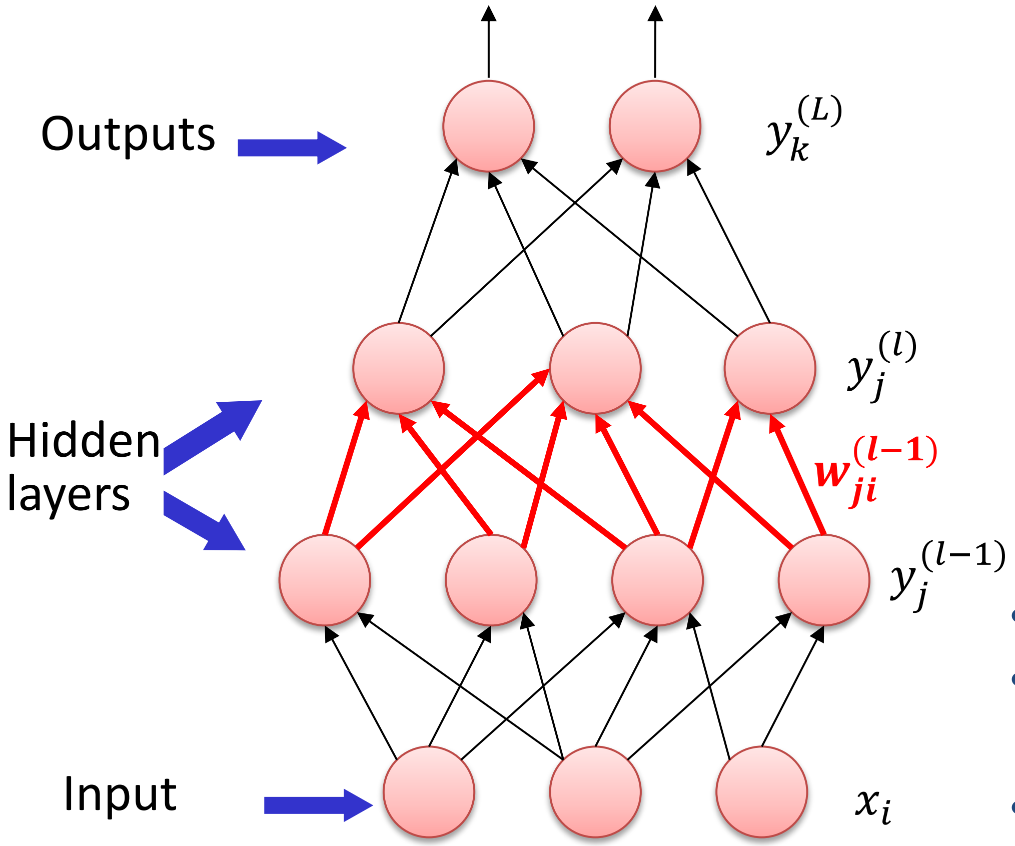
$$f(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

- Rectified linear activation function (ReLU)

$$f(z) = \max(0, x)$$



# Forward pass



For $l = 1, \dots, L-1$ calculate the input to neuron $j$ in the $l$-th layer

$$u_j{}^{(l)} = \sum_i w_{ji}^{(l-1)} y_i^{(l-1)} + b_j^{(l-1)}$$

and its output

$$y_j^{(l)} = f(u_j{}^{(l)})$$

where $f(\cdot)$ is activation function

- Note $y^{(0)} = x$
- There are desired outputs $t_k$ for each input sample
- For $l = L$, $f(\cdot)$ depends on the error function

# Error functions for BP

- Error function $$E = \sum_{n=1}^{N} E^{(n)}$$

  where $E^{(n)}$ is the error function for each input sample $n$

  – Least square error

$$E^{(n)} = \frac{1}{2}\sum_{k=1}^{K}(t_k - y_k^{(L)})^2,\ y_k^{(L)} = \frac{1}{1+\exp(-w_k^{(L-1)\top} y^{(L-1)}) - b_k^{(L-1)})}$$

  – Cross-entropy error

$$E^{(n)} = -\sum_{k=1}^{K} t_k \ln y_k^{(L)},\ \ y_k^{(L)} = \frac{\exp(w_k^{(L-1)\top} y^{(L-1)} + b_k^{(L-1)})}{\sum_{j=1}^{K}\exp(w_j^{(L-1)\top} y^{(L-1)} + b_j^{(L-1)})}$$

  where $t$ is target of the form $(0, 0, \dots, 1, 0, 0)^T$

# Weight adjustment

- Weight adjustment

$$w_{ji}^{(l)} = w_{ji}^{(l)} - \alpha \frac{\partial E}{\partial w_{ji}^{(l)}} \qquad b_j^{(l)} = b_j^{(l)} - \overset{\text{Learning rate}}{\alpha} \frac{\partial E}{\partial b_j^{(l)}}$$

- Weight decay is often used on ${w_{ji}}^{(l)}$ (not necessary on ${b_j}^{(l)}$) which amounts to adding an additional term on the cost function

$$J = E + \frac{\lambda}{2} \sum_{i,j,l} (w_{ji}^{(l)})^2$$

- Weight adjustment on $w$ is changed to

$$w_{ji}^{(l)} = w_{ji}^{(l)} - \alpha \frac{\partial J}{\partial w_{ji}^{(l)}} = w_{ji}^{(l)} - \alpha \frac{\partial E}{\partial w_{ji}^{(l)}} - \alpha \lambda w_{ji}^{(l)}$$

## Backpropagation (BP) algorithm

- The derivative of the loss function is calculated using the chain rule, which has a neat recursive expression if calculated backward
  - This is called BP algorithm (please refer to online materials for details)

## Disadvantages of MLP

- Two many parameters
- Including more layers was not proved to be useful, sometimes even harmful
  - As sigmoid activation functions were usually used, there was “vanishing gradient” effect
  - A two-layer MLP can approximate any function with arbitrary precision!

$\uparrow$

- Unsupervised learning helps to resolve the problem (2006)
- ReLU activation function has attenuated this effect

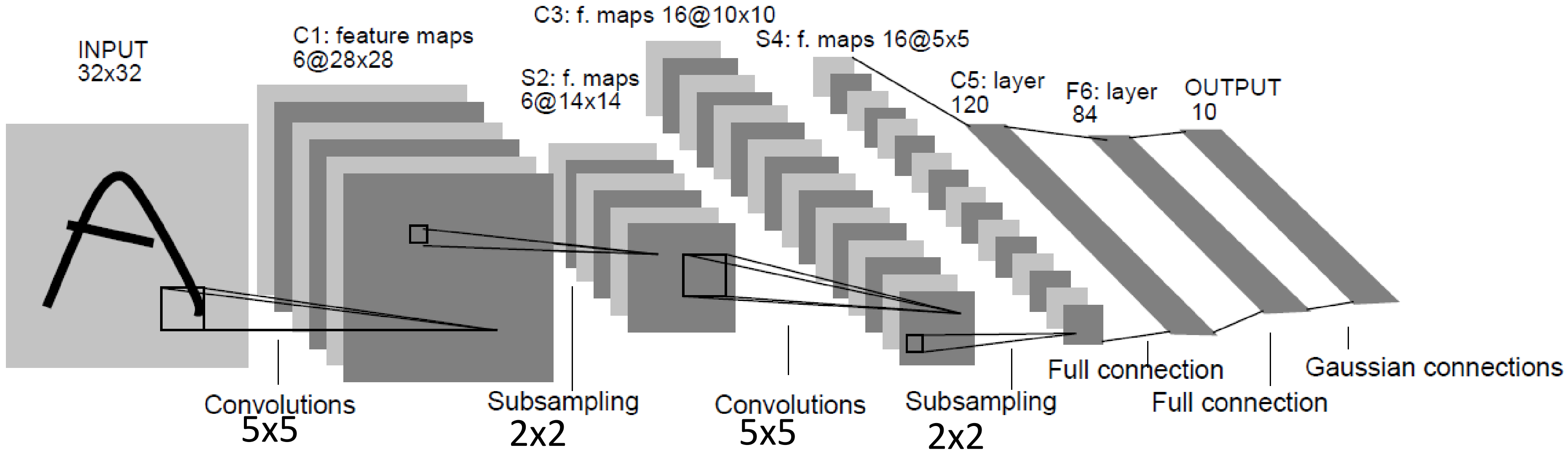
## Outline

- Background
- Multi-layer perceptron
- Convolutional neural nets
- Restricted Boltzmann machine
- Deep belief network
- Applications

Supervised learning

Unsupervised learning

# Convolutional network

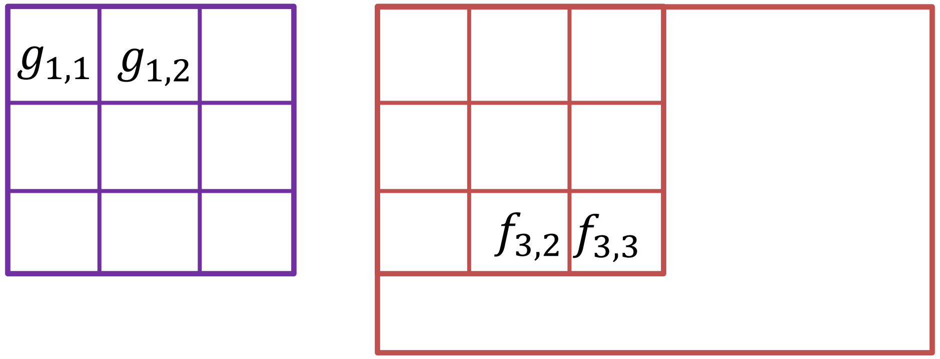


- Local connections and weight sharing
- C layers: convolution
  - Output $y_i = f(\sum_\Omega w_j x_j + b)$ where $\Omega$ is the patch size, $f(\cdot)$ is the sigmoid function, $w$ and $b$ are parameters
- S layers: subsampling (avg pooling)
  - Output $y_i = f\left(\frac{1}{|\Omega|}\sum_\Omega x_j\right)$ where $\Omega$ is the pooling size

# 2D convolution

- Suppose there are two matrices $f$ and $g$ with sizes $M \times N$ and $K_1 \times K_2$, respectively, where $M \geq K_1, N \geq K_2$
- Discrete convolution of the two matrices

$$h[m,n] = (f * g)[m,n] \triangleq \sum_{k_1=1}^{K_1} \sum_{k_2=1}^{K_2} f[m-k_1, n-k_2]g[k_1,k_2]$$



When $m = 4, n = 4$

$$(f * g)_{m,n} = f_{3,3}g_{1,1} + f_{3,2}g_{1,2} + f_{3,1}g_{1,3} + f_{2,3}g_{2,1} + \cdots$$

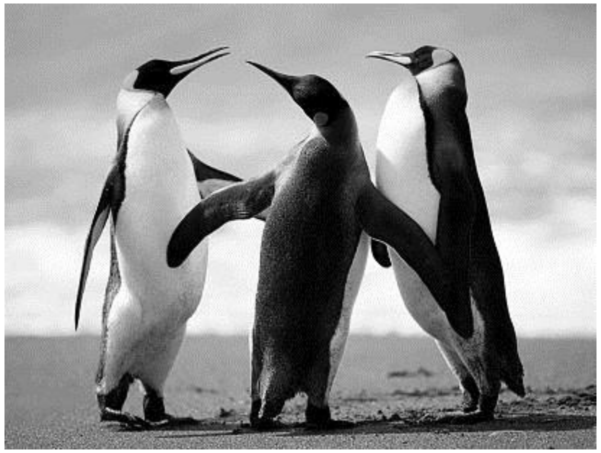
- valid shape: the size of $h$ is $(M - K_1 + 1) \times (N - K_2 + 1)$
- full shape: the size of $h$ is $(M + K_1 - 1) \times (N + K_2 - 1)$
- same shape: the size of $h$ is $M \times N$

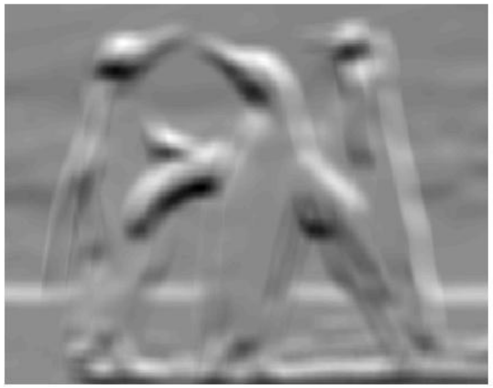
# Example

feature map

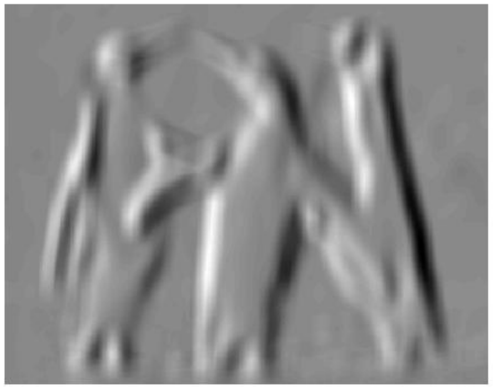
figure

filter



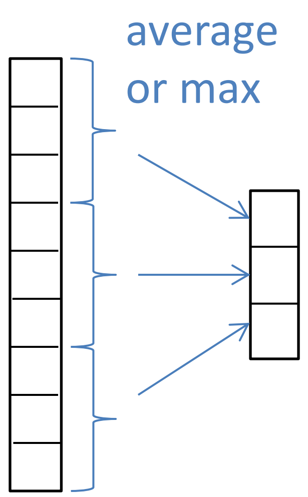


*

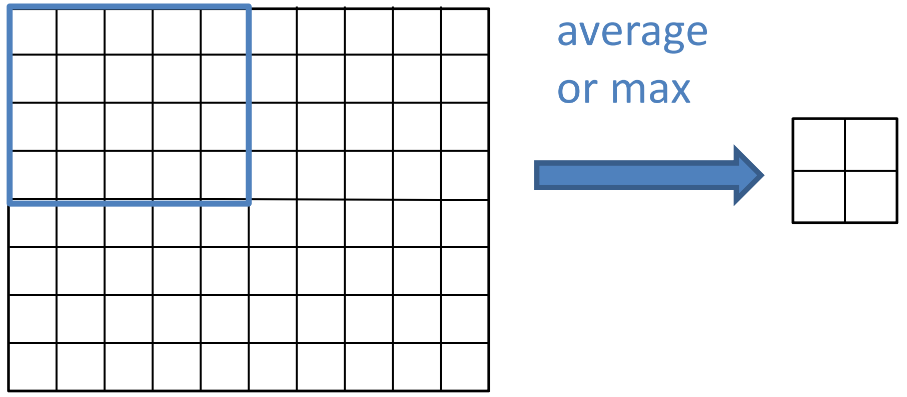


The higher a pixel value (brighter) in the feature map, the more similar between the filter and the corresponding patch in the figure

# Pooling in local regions



poolingsize=3x1



poolingsize=4x5

- Divide the convolved features into *disjoint* $m \times n$ regions, and take the mean (or maximum) feature activation over these regions to obtain the pooled features
- Another often used pooling method is L2 pooling (not discussed in this course)

$$p = \sqrt{\sum_{y_{ij} \in \Omega} y_{ij}^2}$$

## Disadvantage of CNN

- Computationally expensive
- Many researchers including LeCun himself didn’t believe that it could be used for generic pattern recognition

$\uparrow$

GPU helps to resolve this problem

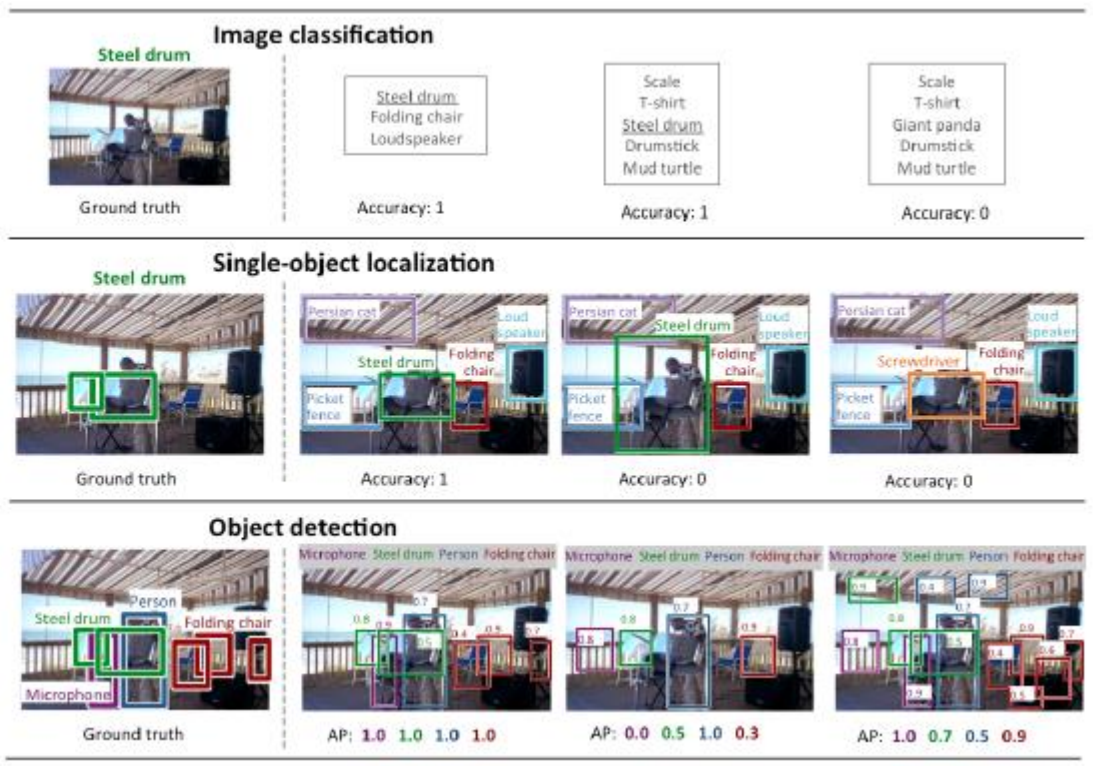
# ImageNet competition (ILSVRC)

Tasks

2010-

2011-

2013-



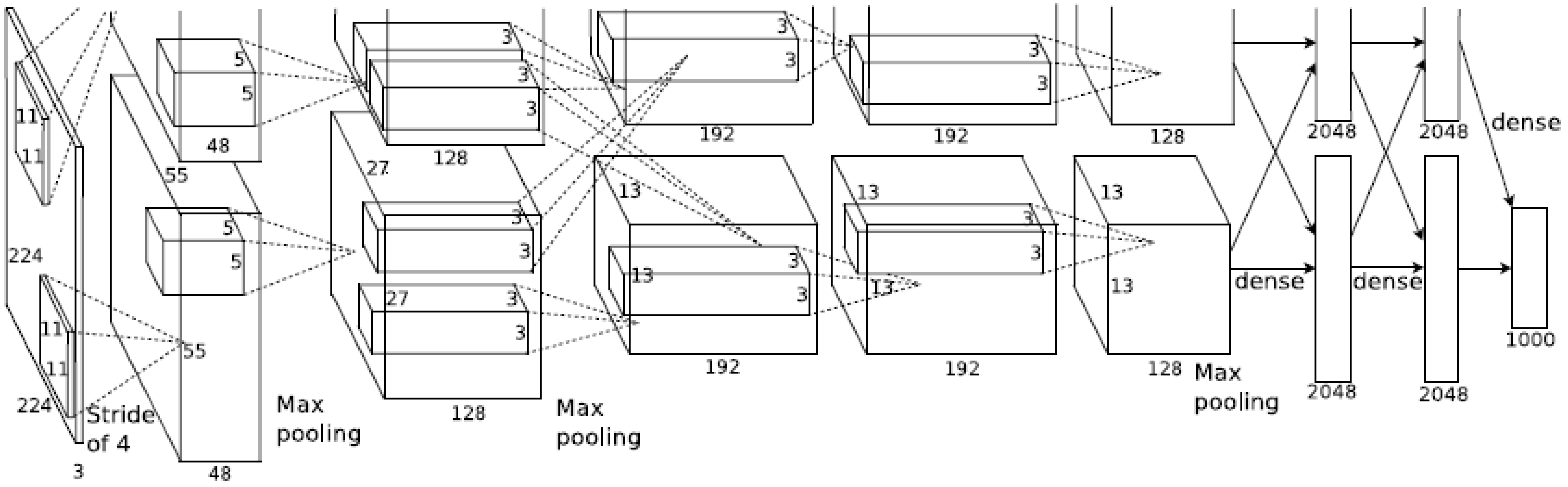
Top-1
Top-5 (preferred)

Two human expert: 5.1%, 12%

The first column shows the ground truth labeling on an example image, and the next three show three sample outputs with the corresponding evaluation score.

Russakovsky, et al., 2014

# CNN’s first show in this competition

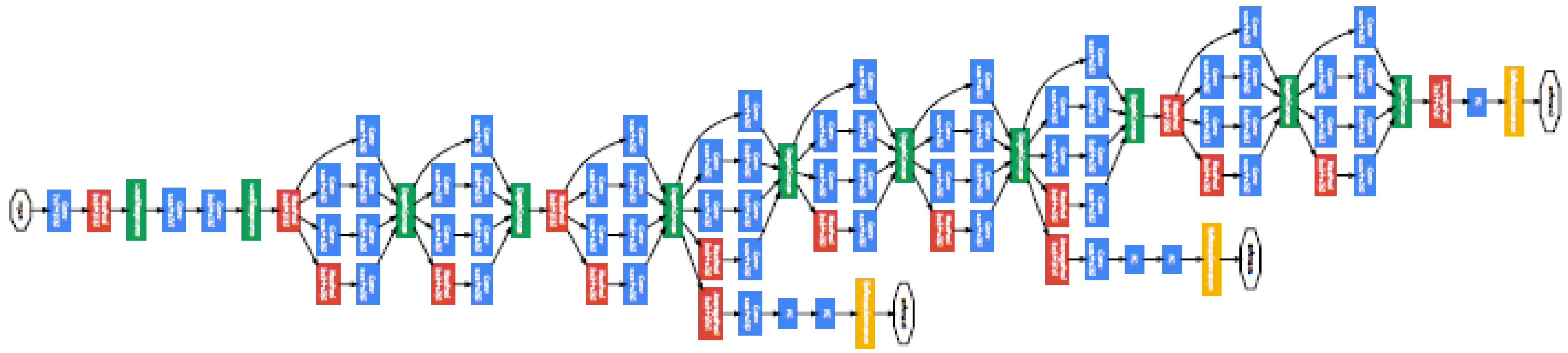


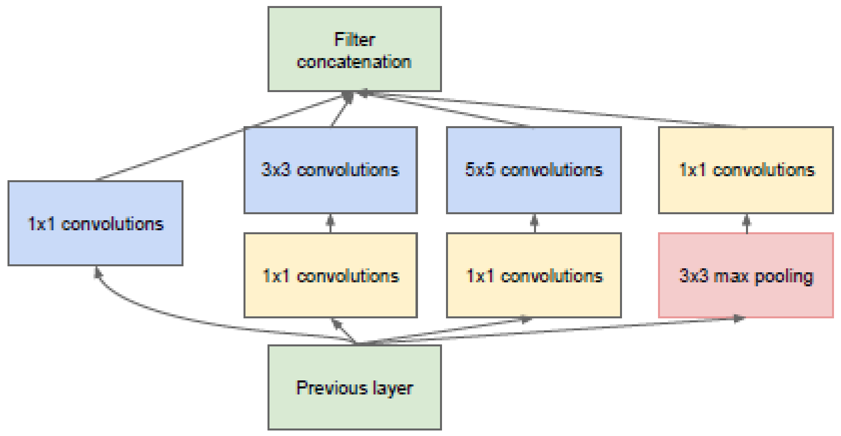
- In total: 60 million parameters

| **Model** | **Top-1** | **Top-5** |
|---|---|---|
| *Sparse coding [2]* | *47.1%* | *28.2%* |
| *SIFT + FVs [24]* | *45.7%* | *25.7%* |
| CNN | **37.5%** | **17.0%** |

Krizhevsky, Sutskever and Hinton, NIPS, 2012

# GoogLeNet





Szegedy, et al., 2014

- 22 weight layers
- Small filters (1x1, 3x3, 5x5)
- Two auxiliary classifiers connected to intermediate layers are used to increase the gradient signal for BP algorithm
- A cpu-based implementation on distributed system

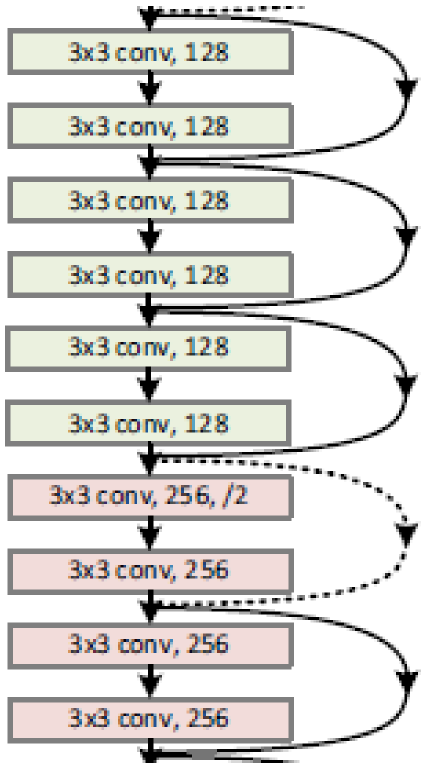
# VGG net

| ConvNet Configuration | | | | | |
|---|---|---|---|---|---|
| A | A-LRN | B | C | D | E |
| 11 weight layers | 11 weight layers | 13 weight layers | 16 weight layers | 16 weight layers | 19 weight layers |
| input (224 × 224 RGB image) | | | | | |
| conv3-64 | conv3-64 | conv3-64 | conv3-64 | conv3-64 | conv3-64 |
| | **LRN** | **conv3-64** | conv3-64 | conv3-64 | conv3-64 |
| maxpool | | | | | |
| conv3-128 | conv3-128 | conv3-128 | conv3-128 | conv3-128 | conv3-128 |
| | | **conv3-128** | conv3-128 | conv3-128 | conv3-128 |
| maxpool | | | | | |
| conv3-256 | conv3-256 | conv3-256 | conv3-256 | conv3-256 | conv3-256 |
| conv3-256 | conv3-256 | conv3-256 | conv3-256 | conv3-256 | conv3-256 |
| | | | **conv1-256** | **conv3-256** | conv3-256 |
| | | | | | **conv3-256** |
| maxpool | | | | | |
| conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 |
| conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 |
| | | | **conv1-512** | **conv3-512** | conv3-512 |
| | | | | | **conv3-512** |
| maxpool | | | | | |
| conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 |
| conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 | conv3-512 |
| | | | **conv1-512** | **conv3-512** | conv3-512 |
| | | | | | **conv3-512** |
| maxpool | | | | | |
| FC-4096 | | | | | |
| FC-4096 | | | | | |
| FC-1000 | | | | | |
| soft-max | | | | | |

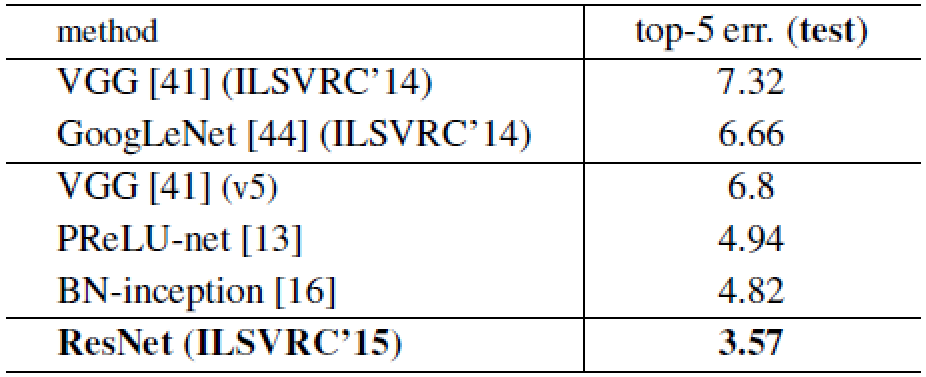
- 3*3 filters are extensively used
- GPU implementation

Simonyan, Zisserman, 2015

# Deep residual network



A 152-layer network achieves 3.57% error rate



| method | top-5 err. (test) |
|---|---|
| VGG [41] (ILSVRC'14) | 7.32 |
| GoogLeNet [44] (ILSVRC'14) | 6.66 |
| VGG [41] (v5) | 6.8 |
| PReLU-net [13] | 4.94 |
| BN-inception [16] | 4.82 |
| **ResNet (ILSVRC'15)** | **3.57** |

1000-layer model has also been tested on CIFAR-10

He et al., 2015

What were people doing between 2006~2012 ?

Unsupervised learning

Since we want the models to be more intelligent and compete with human’s brain, unsupervised learning deserves further investigation

## Outline

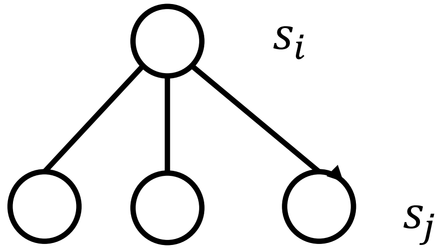
- Background
- Multi-layer perceptron
- Convolutional neural nets
- Restricted Boltzmann machine
- Deep belief network
- Applications

Supervised learning

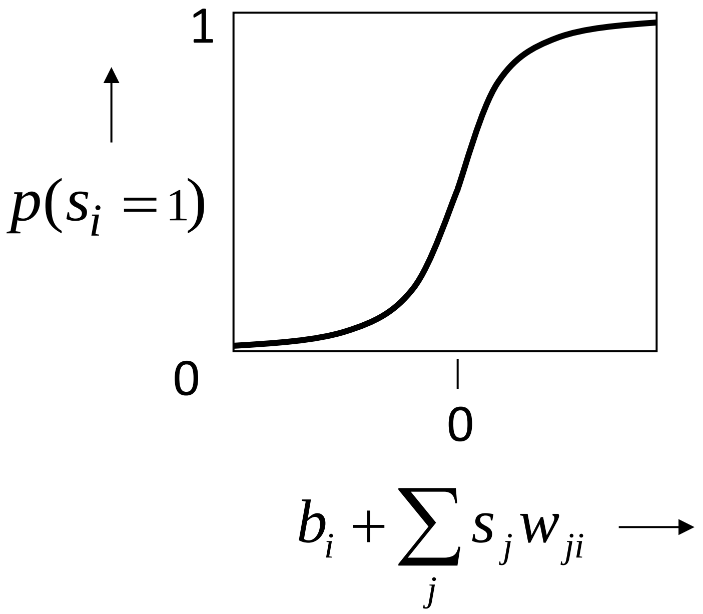
Unsupervised learning

# Stochastic binary units

- Each unit has a state of 0 or 1
- The probability of turning on is determined by

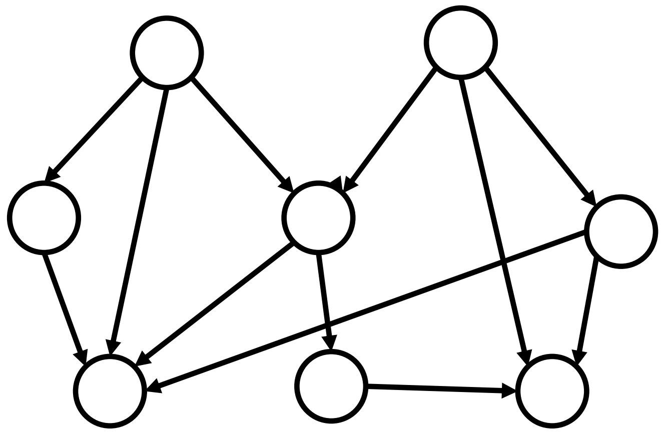


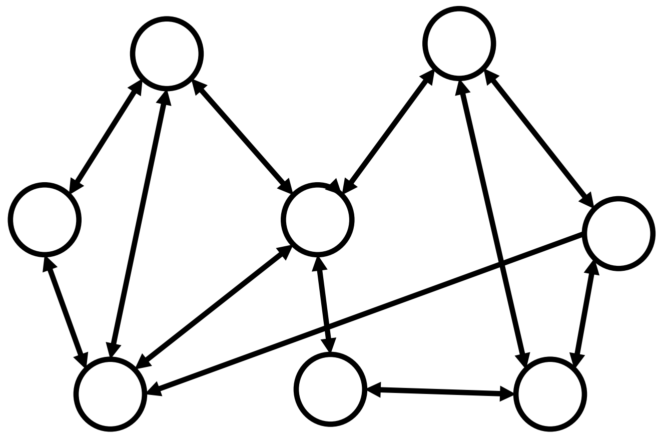
$$p(s_i = 1) \quad = \quad \frac{1}{1 + \exp(-b_i - \sum_j s_j w_{ji})}$$



# Generative models

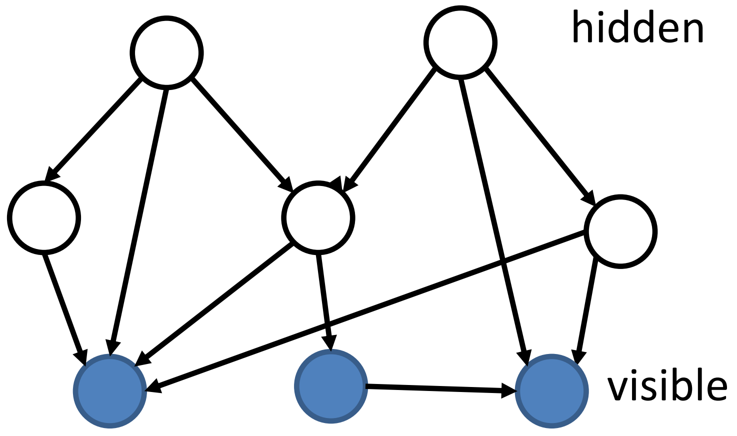
- Directed acyclic graph with stochastic binary units is termed Sigmoid Belief Net
  - Radford Neal 1992
- Undirected graph with stochastic binary units is termed Boltzmann Machine
  - Hinton & Sejnowski, 1983

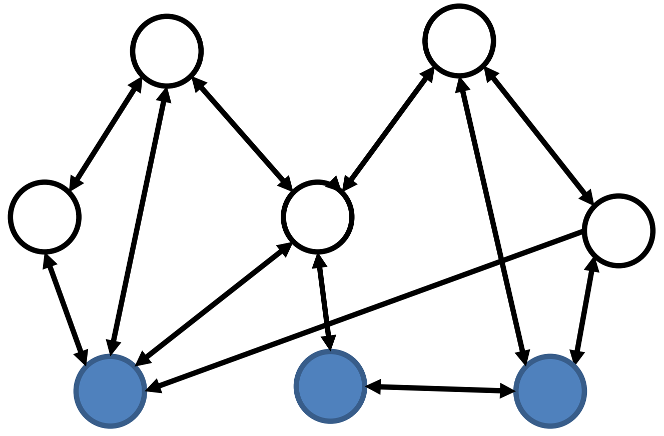




# Generative models

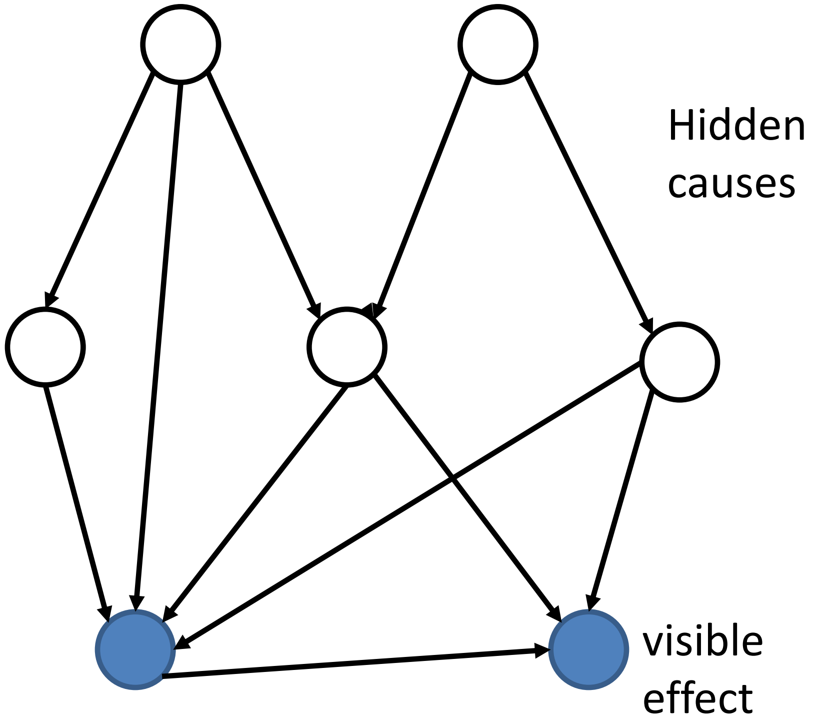
- Learning: Adjust the interactions between variables to make the network more likely to generate the observed data
- Inference: Infer the states of the unobserved variables
- Generate: Generate the observed data





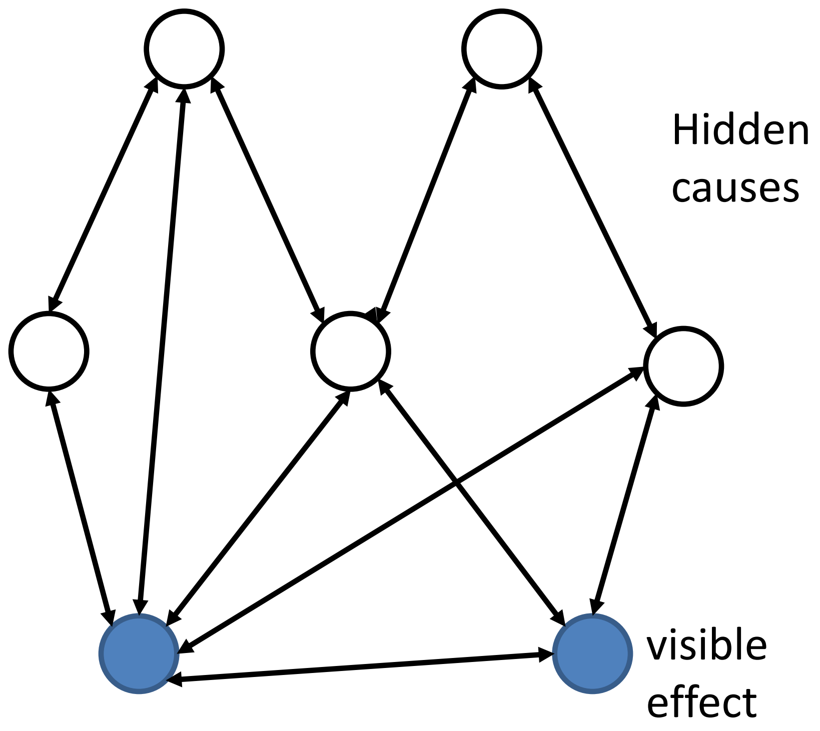
# Learning deep belief nets

- Easy to generate an unbiased example at the leaf nodes
- Hard to infer the posterior distribution over all possible configurations of hidden causes
  - Hard to even get a sample from the posterior



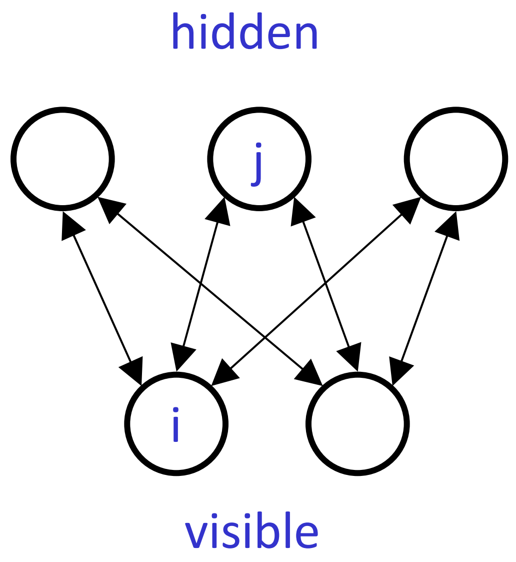
# Learning Boltzmann machine

- Hard to generate an unbiased example for the visible units
- Hard to infer the posterior distribution over all possible configurations of hidden causes
  - Hard to even get a sample from the posterior

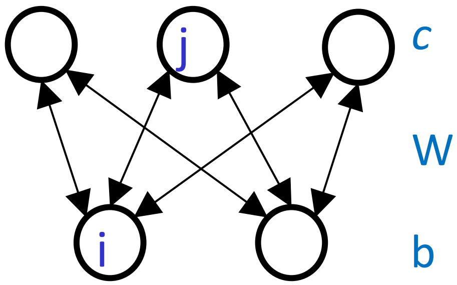


# Restricted Boltzmann machines

- Restrict the connectivity to make learning easier.
  - Only one layer of hidden units.
  - No connections between hidden units.
  - Every unit can take only 1 or 0 stochastically
- In an RBM, the hidden units are conditionally independent given the visible states.
  - We can quickly get an unbiased sample from the posterior distribution when given a data-vector.



# Energy model



- Joint distribution

partition function

$$P[v,h] = \frac{\exp(-E(v,h))}{Z} \quad \text{where} \quad Z = \sum_{v,h} \exp(-E(v,h)).$$

- The energy function

$$E(v,h) = -v \cdot W \cdot h - b \cdot v - c \cdot h$$

- The probability distribution of data

$$P[v;\mathcal{G}] = \sum_h P[v,h;\mathcal{G}] = \frac{1}{Z}\sum_h \exp(-E(v,h)).$$

where $\mathcal{G} \triangleq (W,b,c)$

## Maximum data log likelihood

- The primary goal

$$P[v;\mathcal{G}] = \frac{1}{Z}\sum_h \exp(-E(v,h)).$$

$$\mathcal{G}^* = \arg\max\langle \ln P[v;\mathcal{G}]\rangle$$

- The gradient

$$E(v,h) = -v\cdot W\cdot h - b\cdot v - c\cdot h$$

$$\frac{\partial \ln P[v;\mathcal{G}]}{\partial W_{ij}} = \frac{\partial}{\partial W_{ij}}\left(\ln\sum_h \exp(-E(v,h)) - \ln\sum_{v,h}\exp(-E(v,h))\right)$$

$$= \sum_h \frac{\exp(-E)}{\sum_h \exp(-E)} v_i h_j - \sum_{v,h}\frac{\exp(-E)}{\sum_{v,h}\exp(-E)} v_i h_j$$

$$= \sum_h P[h|v;\mathcal{G}]h_j v_i - \sum_{v,h} P[v,h;\mathcal{G}]h_j v_i.$$

$$\frac{\partial \ln P[v;\mathcal{G}]}{\partial b_i} = \sum_h P[h|v;\mathcal{G}]v_i - \sum_{v,h}P[v,h;\mathcal{G}]v_i$$

$$\frac{\partial \ln P[v;\mathcal{G}]}{\partial c_j} = \sum_h P[h|v;\mathcal{G}]h_j - \sum_{v,h}P[v,h;\mathcal{G}]h_j$$

Approximate avg with one sample

# Learning rule

- Stochastic gradient ascent (*n* is the sample index)

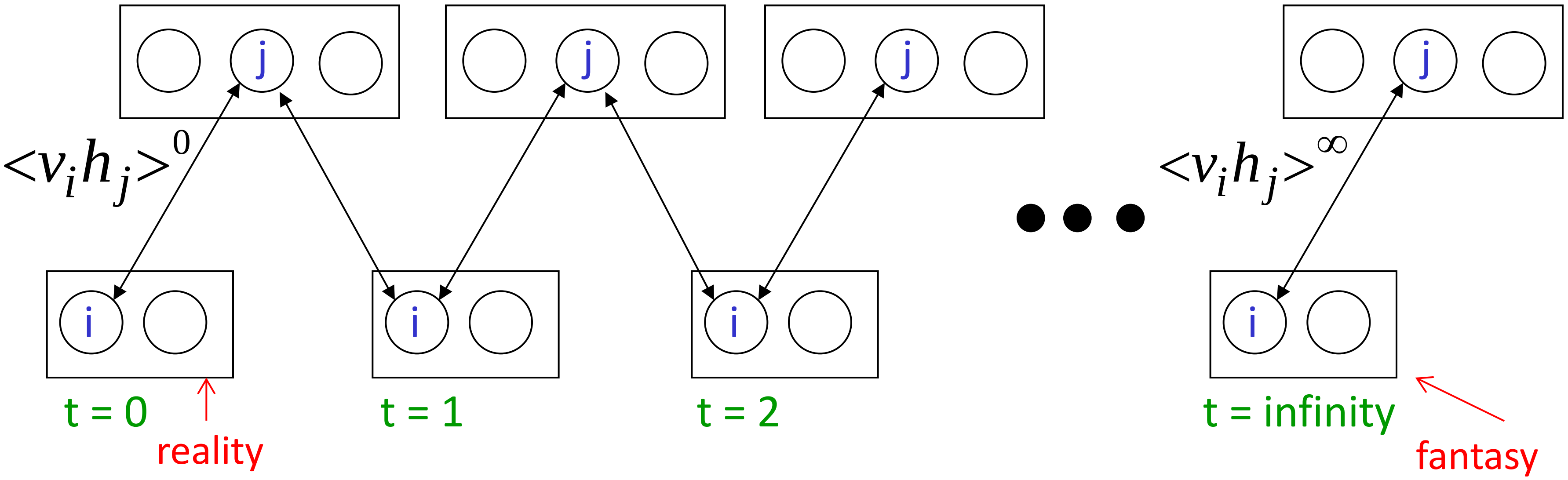
$$
\begin{aligned}
W_{ij} &= W_{ij} + \epsilon_W(h_j(v^n)v_i^n - h_j(t \to \infty)v_i(t \to \infty)) \\
b_i &= b_i + \epsilon_b(v_i^n - v_i(t \to \infty)) \\
c_j &= c_j + \epsilon_c(h_j(v^n) - h_j(t \to \infty))
\end{aligned}
$$

- Wake phase: Gibbs sampling is used to calculate $h(v^n)$
- Sleep phase: Gibbs sampling is used to calculate $h(t \to \infty$ and $v(t \to \infty)$

What we only need are $P(h_j = 1|v)$ and $P(v_i = 1|h)$, which are given as sigmoid functions

# Illustration of learning

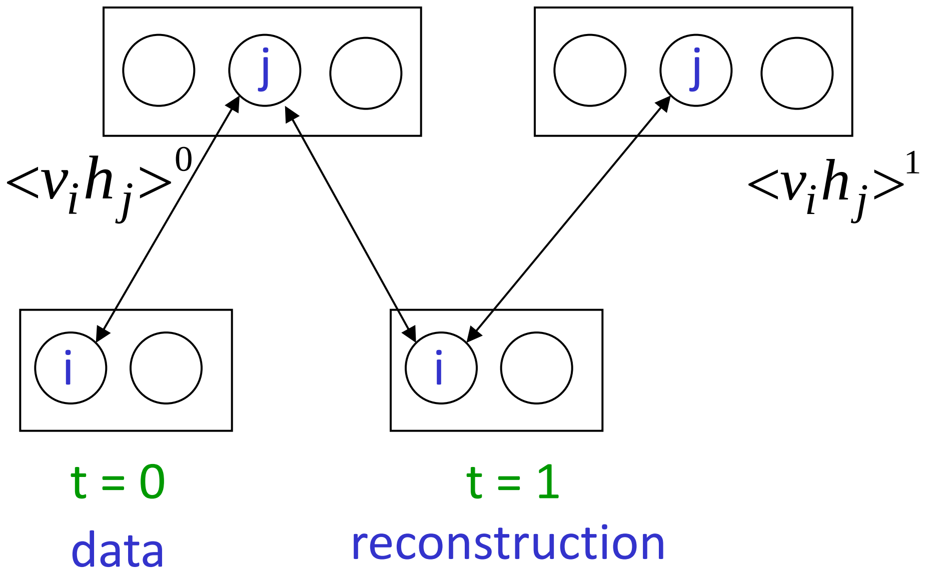
$$W_{ij} = W_{ij} + \epsilon_W(\langle h_j v_i \rangle^0 - \langle h_j v_i \rangle^\infty)$$
$$b_i = b_i + \epsilon_b(\langle v_i \rangle - \langle v_i \rangle^\infty)$$
$$c_j = c_j + \epsilon_c(\langle h_j \rangle - \langle h_j \rangle^\infty)$$



Alternate between updating all the hidden units in parallel and updating all the visible units in parallel.

# Contrastive divergence learning

- CD-1
  - Start with a training vector on the visible units.
  - Update all the hidden units in parallel
  - Update the all the visible units in parallel to get a “reconstruction”
  - Update the hidden units again
- CD-*n*
  - Keep running for *n* steps



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

$$\Delta w_{ij} = \varepsilon \, ( <v_i h_j>^0 - <v_i h_j>^n )$$

## Outline

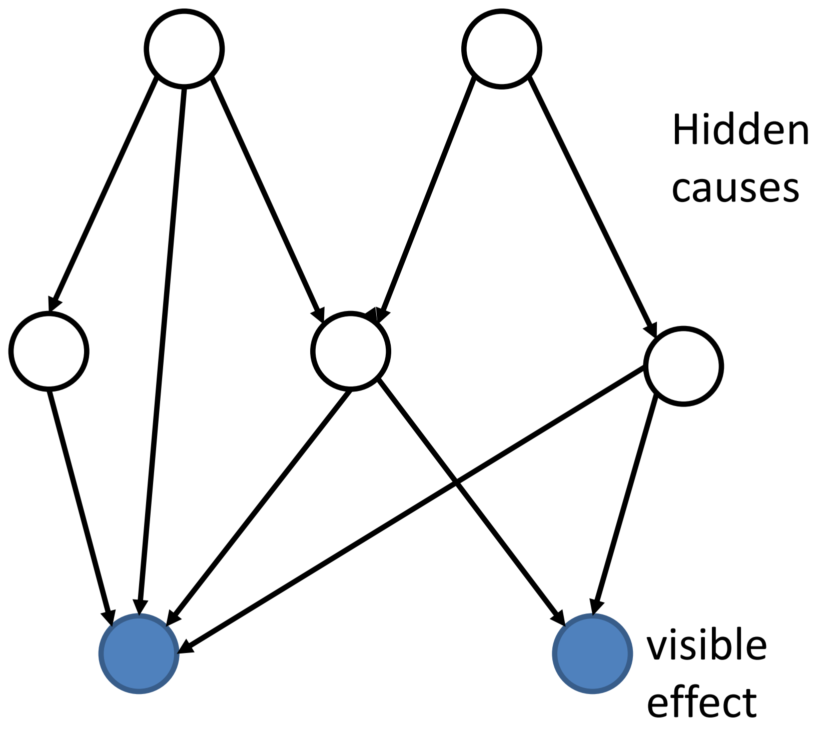
- Background
- Multi-layer perceptron
- Convolutional neural nets
- Restricted Boltzmann machine
- Deep belief network
- Applications

$\left.\begin{array}{l}\text{Multi-layer perceptron}\\\text{Convolutional neural nets}\end{array}\right\}$ Supervised learning

$\left.\begin{array}{l}\text{Restricted Boltzmann machine}\\\text{Deep belief network}\end{array}\right\}$ Unsupervised learning

# Belief networks

- Easy to generate an unbiased example at the leaf nodes
- Hard to infer the posterior distribution over all possible configurations of hidden causes
- So how can we learn deep belief nets that have millions of parameters?



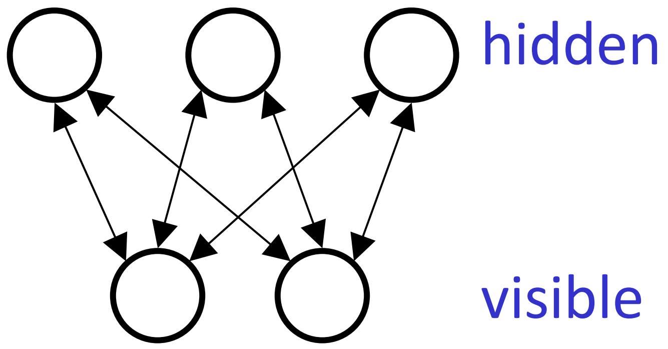
$$p(s_i = 1) = \frac{1}{1 + \exp(-b_i - \sum_j s_j w_{ji})}$$

## Some methods for learning deep belief nets

- Monte Carlo methods can be used to sample from the posterior.
  - But its painfully slow for large, deep models.
- In the 1990's people developed variational methods for learning deep belief nets
  - These only get approximate samples from the posterior.
  - Nevertheless, the learning is still guaranteed to improve a variational  bound on the log probability of generating the observed data.

## The breakthrough

- To learn deep nets efficiently, we need to learn one layer of features at a time.
- We need a way of learning one layer at a time that takes into account the fact that we will be learning more hidden layers later.
  - We solve this problem by using RBM.



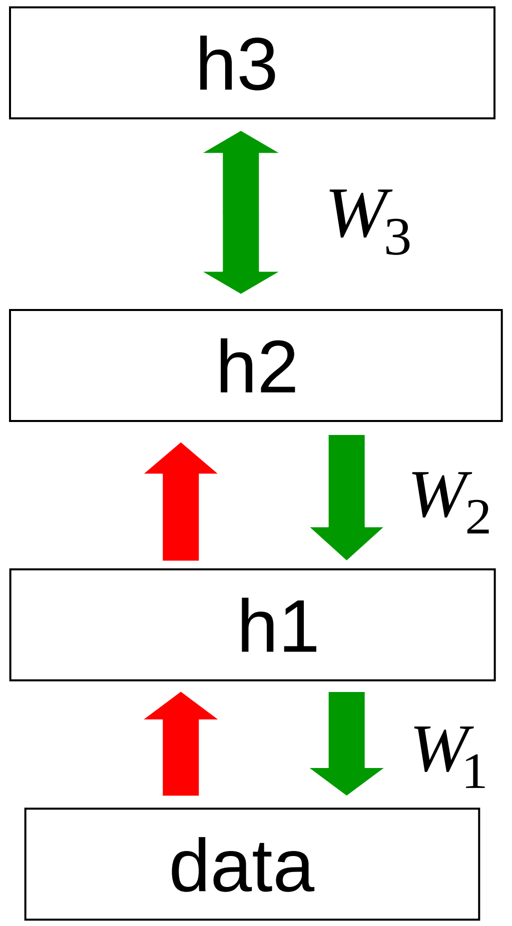
## Training a deep network

- Stacking RBMs to form deep architecture
  - First train an RBM that receives input directly from the pixels
  - Then treat the activations of the hidden layer as if they were pixels and train a second hidden layer
  - Repeat the process
- Each time we add another layer of features we improve a variational lower bound on the log probability of the training data
  - Proof is a little bit complicated

# The generative model after learning a 3-layer model

- To generate data:
  1. Get an equilibrium sample from the top-level RBM by performing alternating Gibbs sampling for a long time.
  2. Perform a top-down pass to get states for all the other layers.

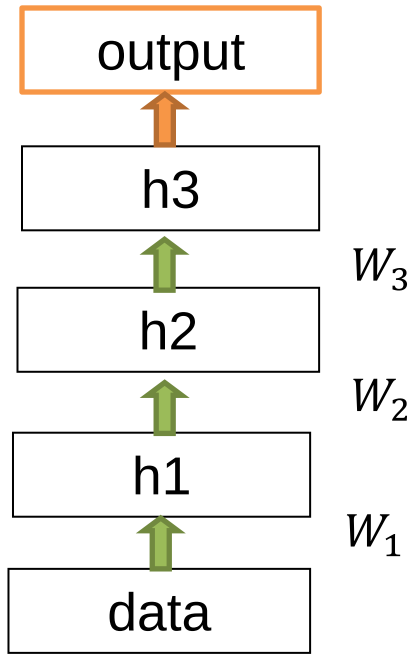
So the bottom-up connections are not part of the generative model. They are just used for inference.



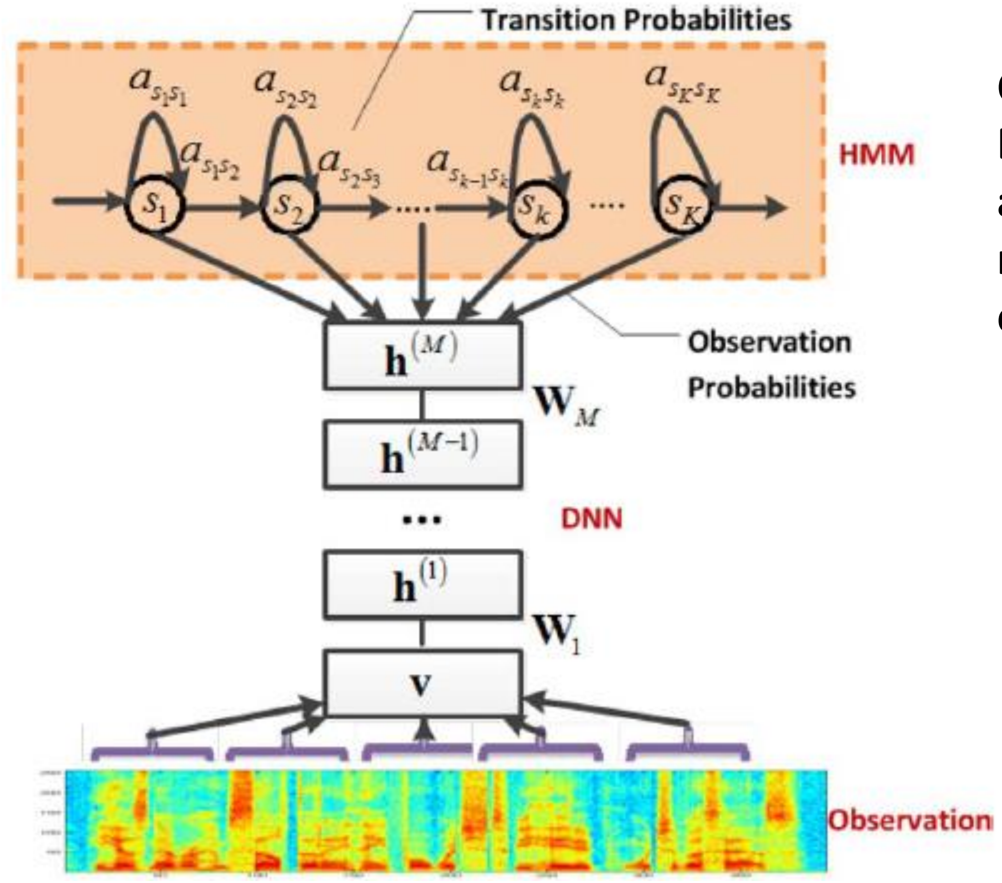
## How to use the pre-trained DBN?

## Method 1: Add a layer on top

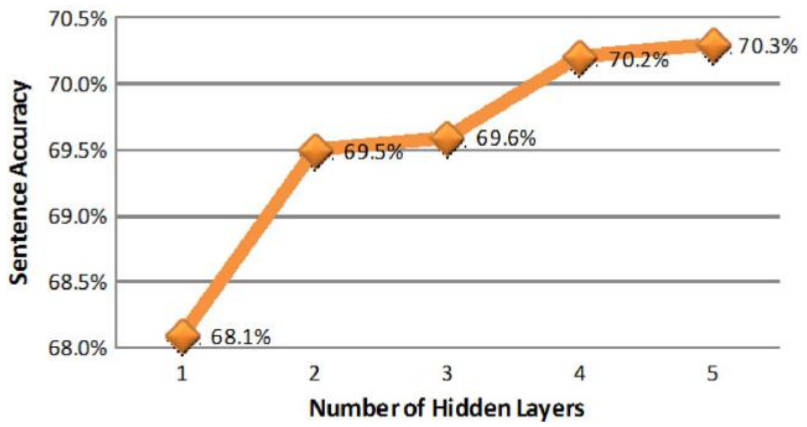
- Add a softmax layer on top, then perform BP training with the pretrained weights as initial weights
  - Dahl, Yu, Deng, Acero, IEEE TASLP, 2012
- Add an SVM on top
  - Lee, et al, ICML 2009



# Speech Recognition



**Compared with CD-GMM-HMMs， CD-DNN-HMMs improved 5.8% and 9.2% accuracy using the minimum phone error rate (MPE) and maximum-likelihood (ML) criteria**

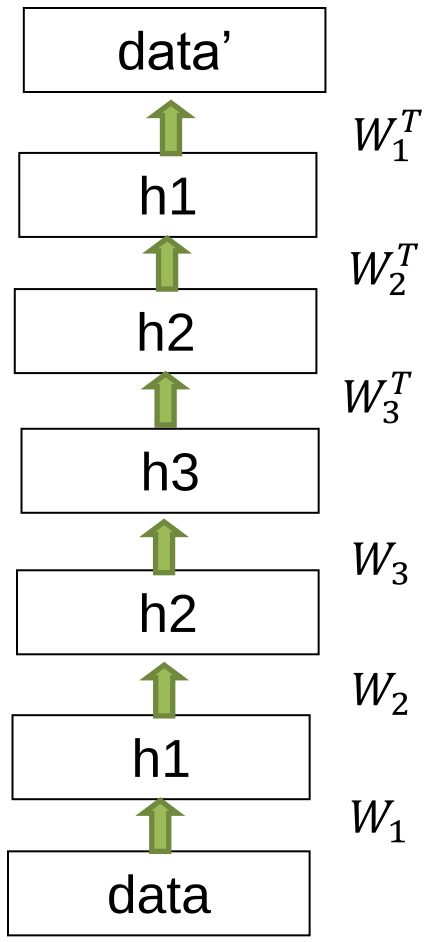


# Speech Translation by Microsoft Research

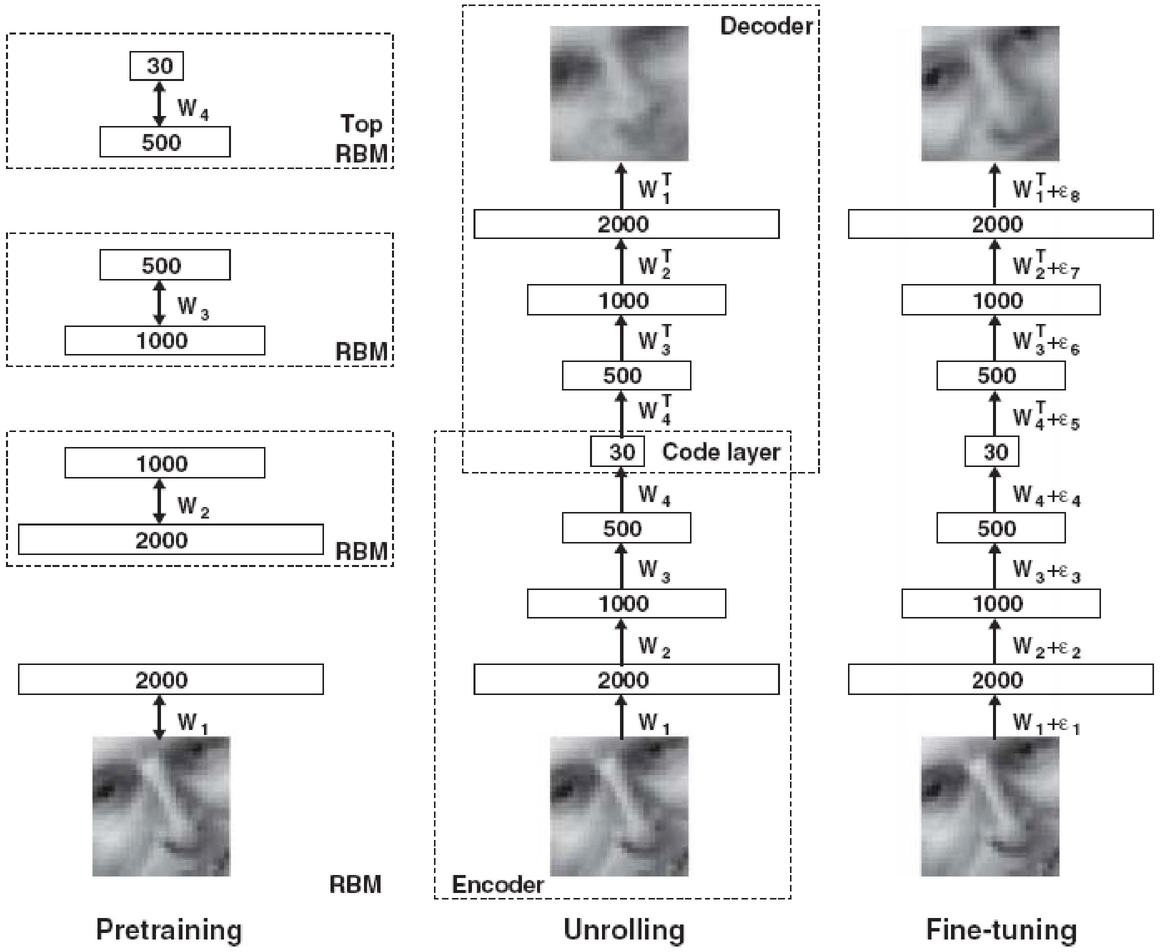
- See youku: http://v.youku.com/v_show/id_XNDc0MDY4ODI0.html

# Method 2: Unroll the architecture and fine-tune with BP

- The target is the data itself
- If the number of units in layer h3 is small, then it performs data compression
  - Hinton, Salakhutdinov, Science, 2006

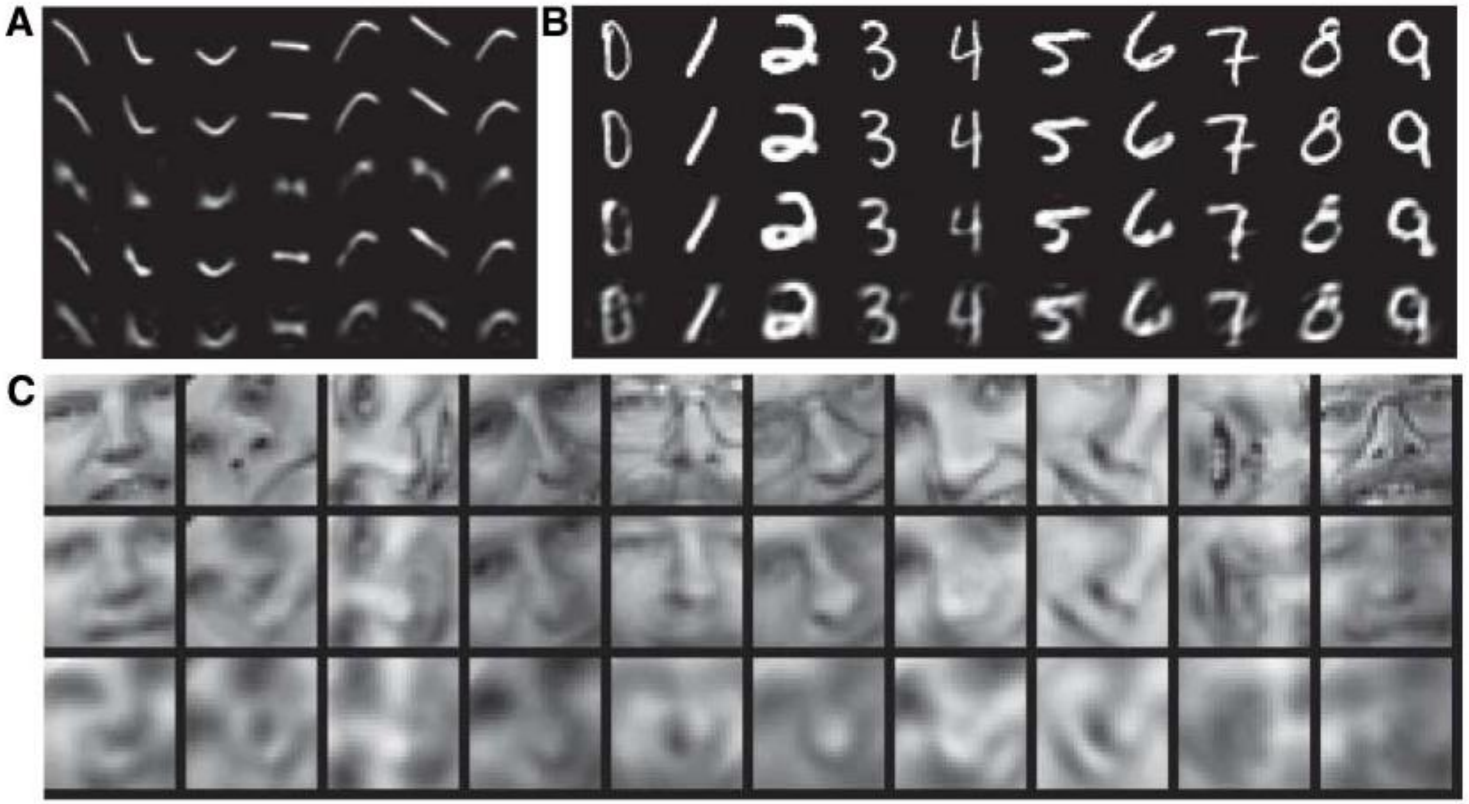


# Data compression



Hinton, Salakhutdinov, Science, 2006

# Reconstruction Results



real data

30-D deep auto

30-D logistic PCA

30-D PCA

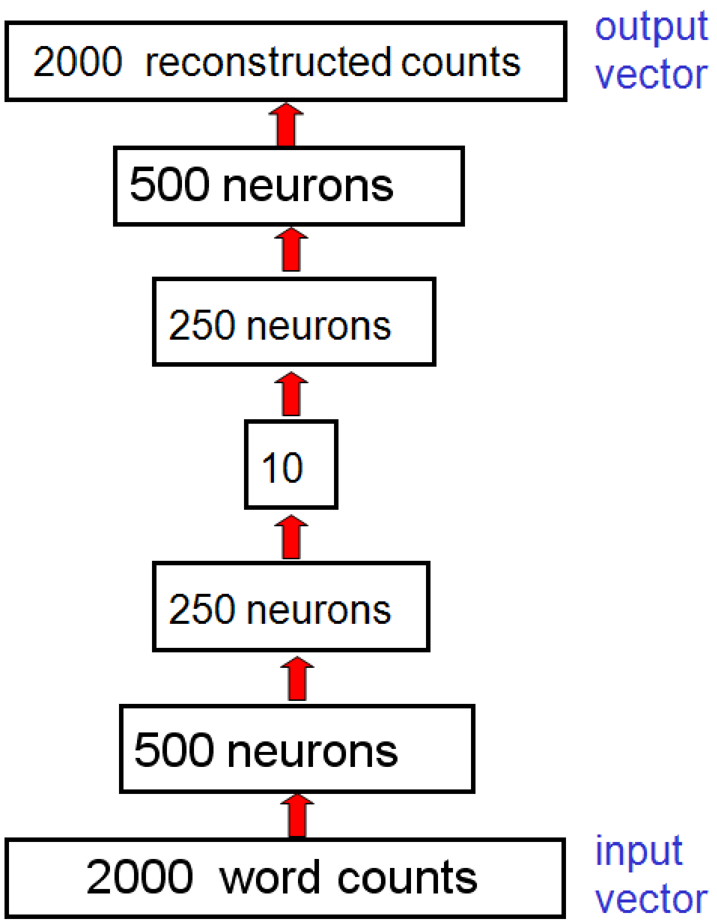
real data

30-D deep auto

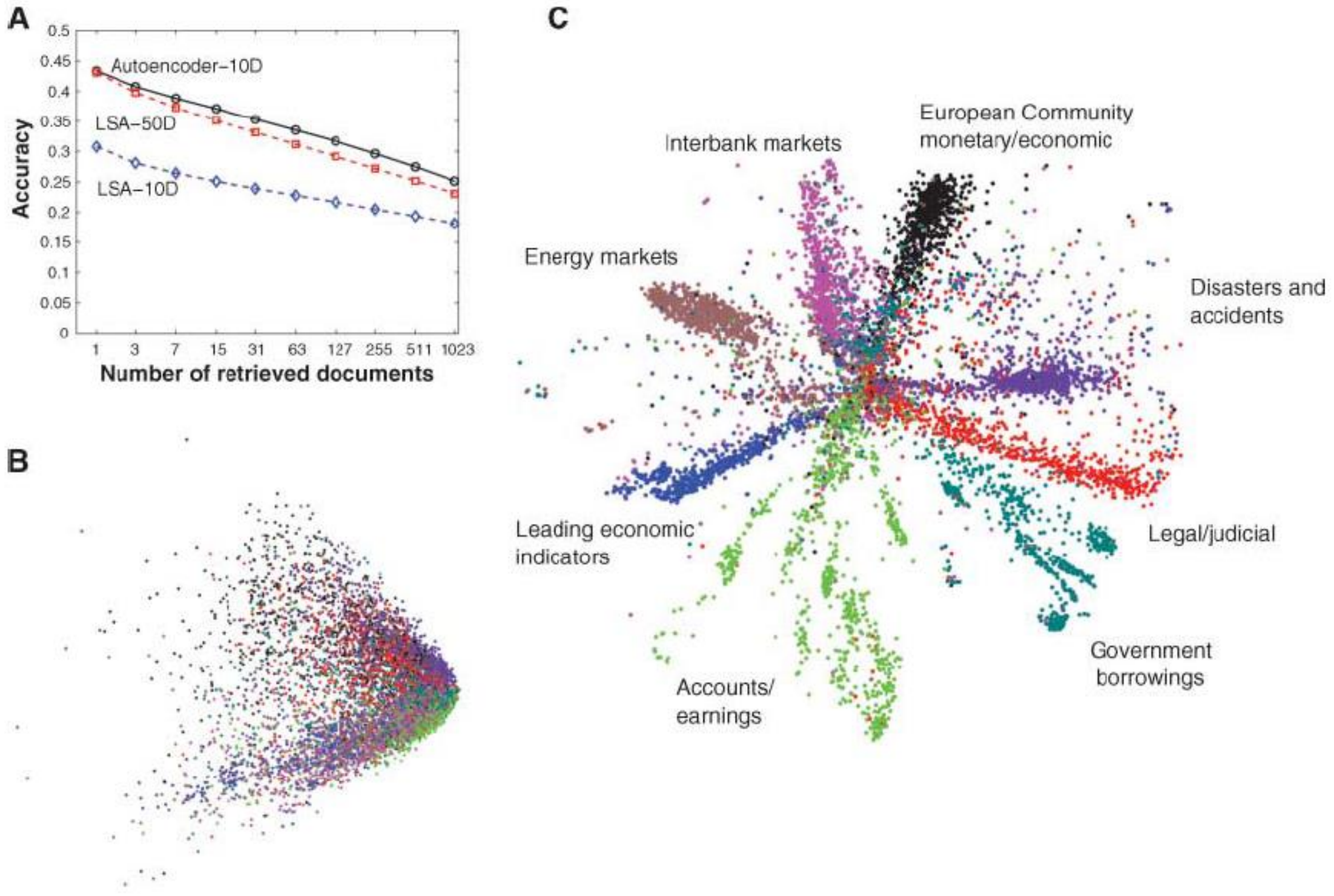
30-D PCA

# Retrieving Documents

- Convert each document into a "bag of words".
  - This a 2000D vector
- Compress them to 10D vectors
- Compare documents based on these 10D vectors



# Results on 804,414 Newswire Stories



## Outline

- Background
- Multi-layer perceptron
- Convolutional neural nets
- Restricted Boltzmann machine
- Deep belief network
- Applications

Supervised learning

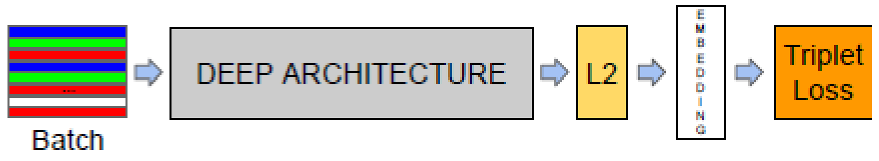
Unsupervised learning

# 人脸验证

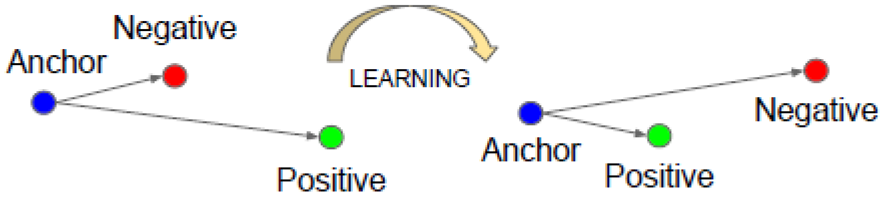
| Model | Accuracy (%) |
|---|---|
| DeepFace (2014) | 97.25 |
| DeepID (2014) | 97.45 |
| DeepID2 (2014) | 99.15 |
| DeepID2+ (2014) | 99.47 |
| DeepID3 (2014) | 99.53 |
| FaceNet (2015) | 99.63 |

# FaceNet

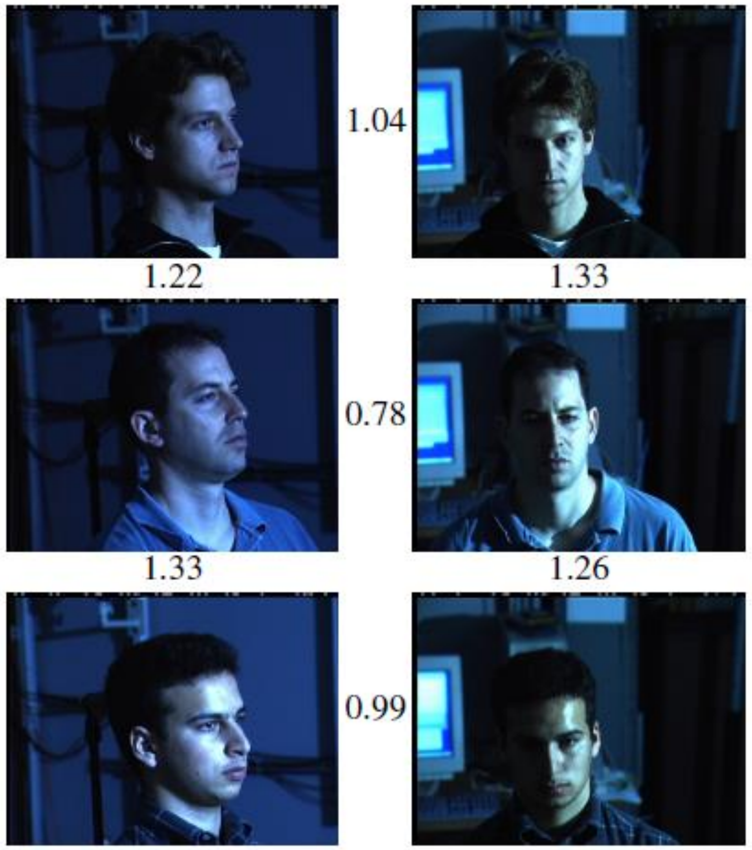
- Architecture



- Triplet loss

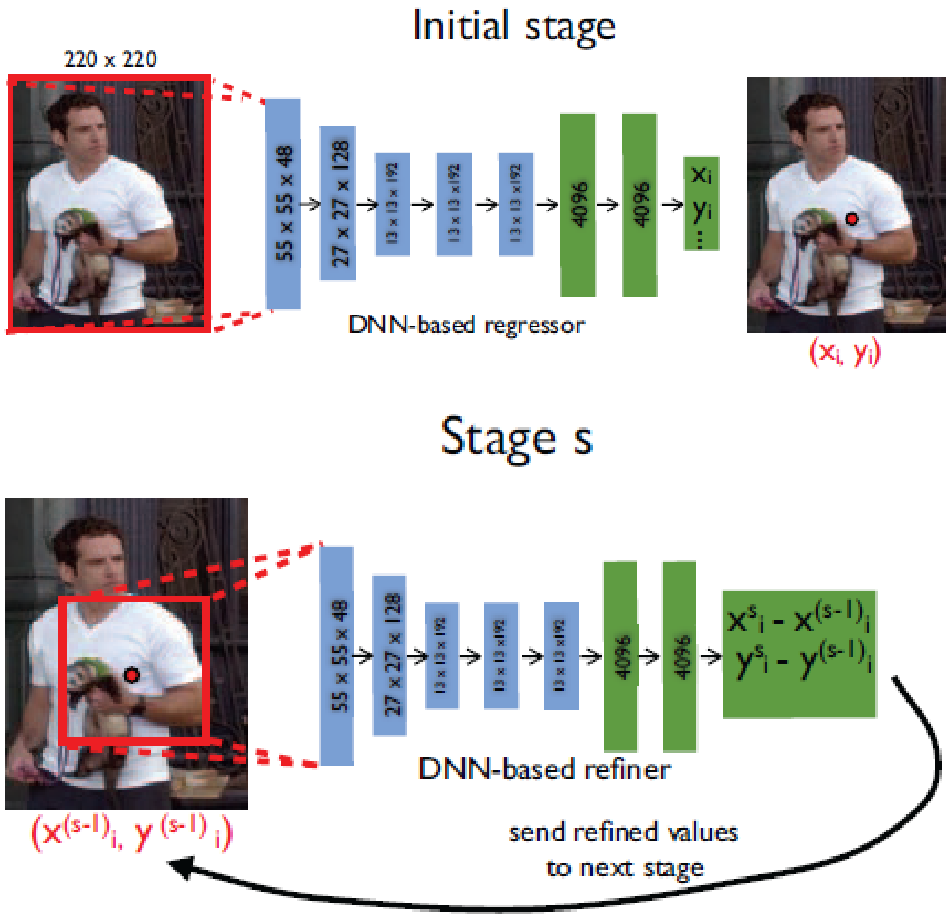


- 100M-200M training faces of about 8M different identities



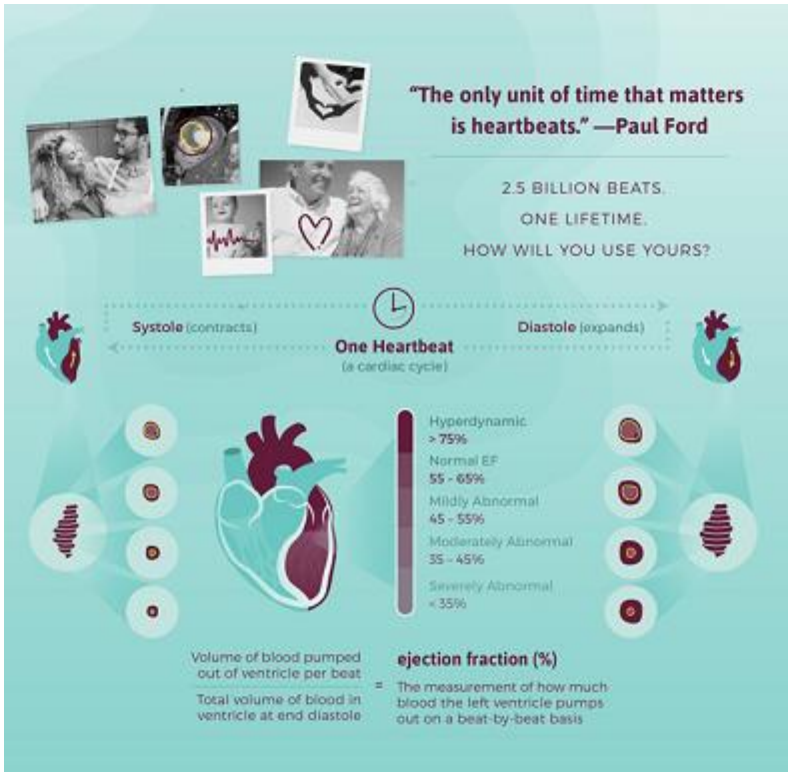
# 人的姿态估计

CNN as a regressor

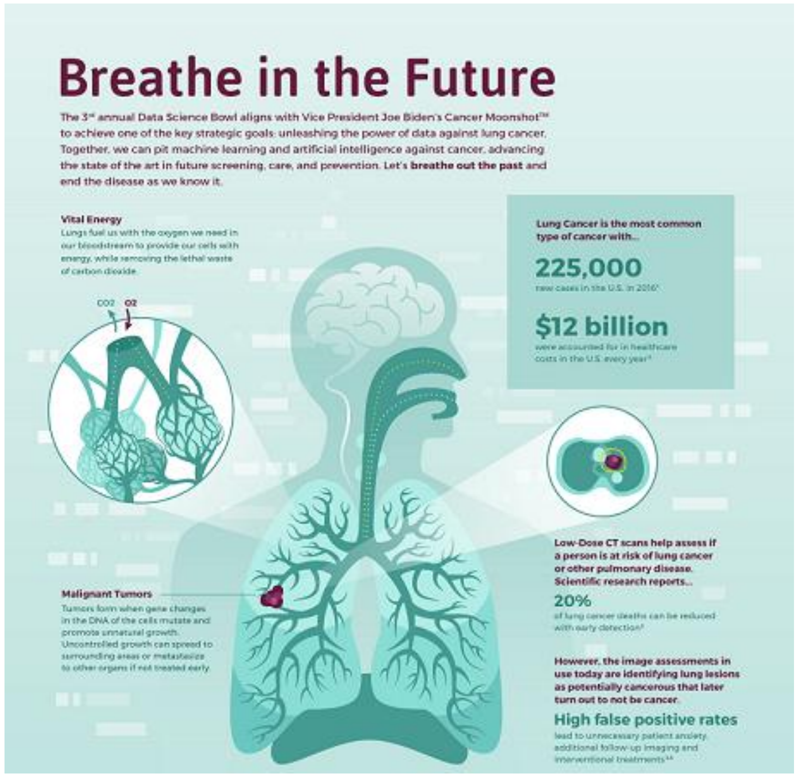


Toshev, Szegedy, 2014

# 医学图像处理


"The only unit of time that matters is heartbeats." —Paul Ford
2.5 BILLION BEATS.
ONE LIFETIME.
HOW WILL YOU USE YOURS?
Systole (contracts)
One Heartbeat
(a cardiac cycle)
Diastole (expands)
Hyperdynamic > 75%
Normal EF 55 - 65%
Mildly Abnormal 45 - 55%
Moderately Abnormal 35 - 45%
Severely Abnormal < 35%
Volume of blood pumped out of ventricle per beat
Total volume of blood in ventricle at end diastole
=
ejection fraction (%)
The measurement of how much blood the left ventricle pumps out on a beat-by-beat basis

Kaggle Data Science Bowl 2016

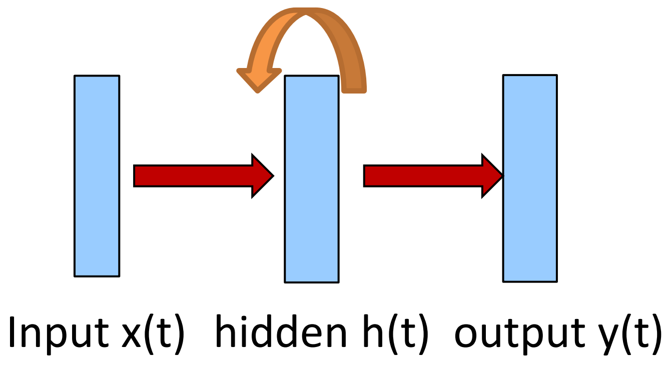

Breathe in the Future
The 3rd annual Data Science Bowl aligns with Vice President Joe Biden's Cancer Moonshot™ to achieve one of the key strategic goals: unleashing the power of data against lung cancer. Together, we can pit machine learning and artificial intelligence against cancer, advancing the state of the art in future screening, care, and prevention. Let's breathe out the past and end the disease as we know it.
Vital Energy
Lungs fuel us with the oxygen we need in our bloodstream to provide our cells with energy, while removing the lethal waste of carbon dioxide.
CO2
O2
Lung Cancer is the most common type of cancer with...
225,000
new cases in the U.S. in 2016
$12 billion
were accounted for in healthcare costs in the U.S. every year
Malignant Tumors
Tumors form when gene changes in the DNA of the cells mutate and promote unnatural growth. Uncontrolled growth can spread to surrounding areas or metastasize to other organs if not treated early.
Low-Dose CT scans help assess if a person is at risk of lung cancer or other pulmonary disease. Scientific research reports...
20%
of lung cancer deaths can be reduced with early detection
However, the image assessments in use today are identifying lung lesions as potentially cancerous that later turn out to not be cancer.
High false positive rates
lead to unnecessary patient anxiety, additional follow-up imaging and interventional treatments

Kaggle Data Science Bowl 2017

# 反馈神经网络用于语音识别

- The simple RNN (The Elman network)

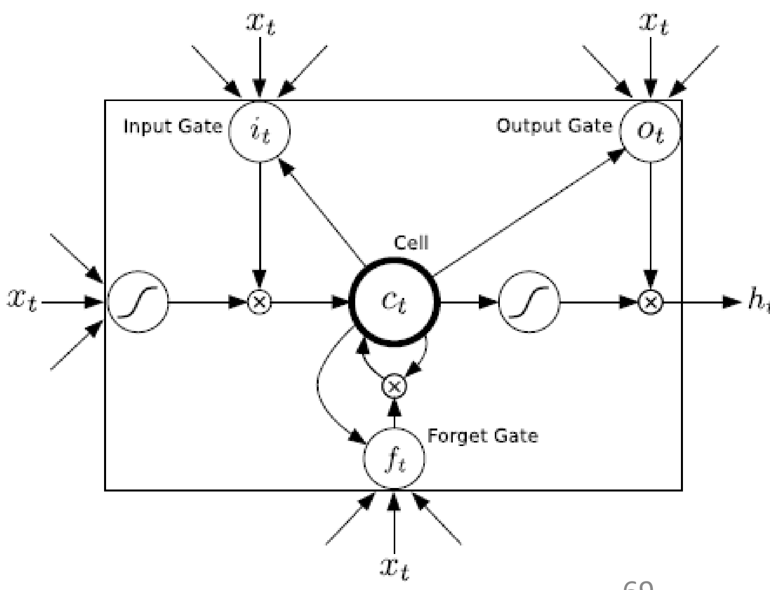
$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right)$$
$$y_t = W_{hy}h_t + b_y$$



- The long short-term memory (LSTM) model implement a complicated *H* function

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right)$$
$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right)$$
$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right)$$
$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right)$$
$$h_t = o_t \tanh(c_t)$$

A memory cell

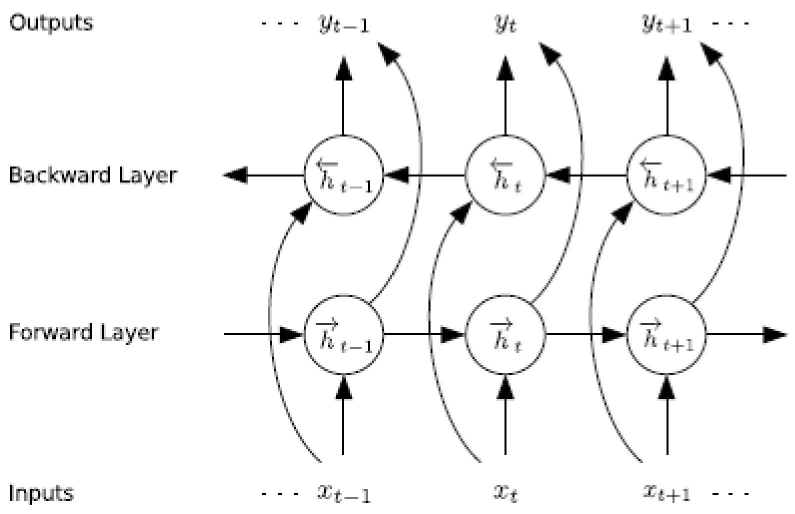


# 反馈神经网络用于语音识别

- Bidirectional RNN

$$\overrightarrow{h}_t = \mathcal{H}\left(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right)$$
$$\overleftarrow{h}_t = \mathcal{H}\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}\right)$$
$$y_t = W_{\overrightarrow{h}y}\overrightarrow{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y$$

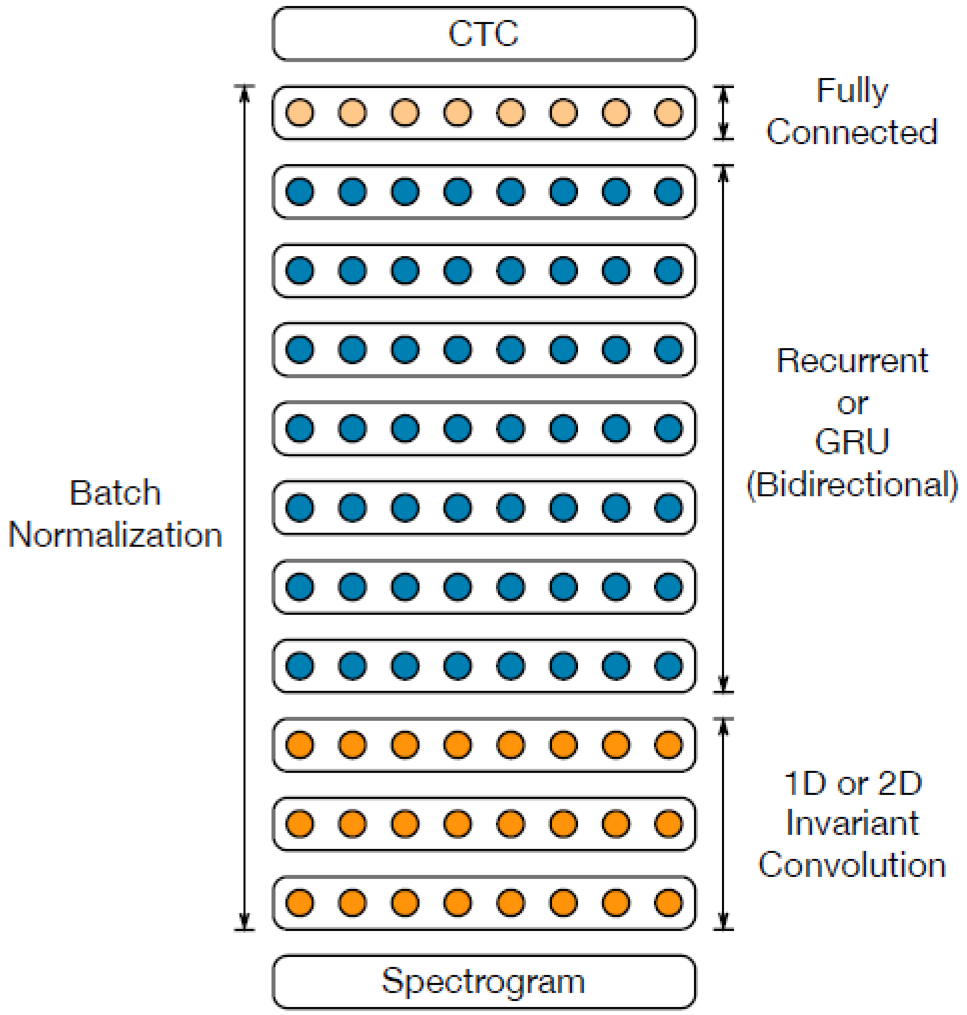
- Combing BRNNs with LSTM gives bidirectional LSTM
- Softmax is used in the output layer for every time t (size K+1)
  - K phonemes plus one non-output unit



Graves et al., 2013

State-of-the-art accuracy on TIMIT

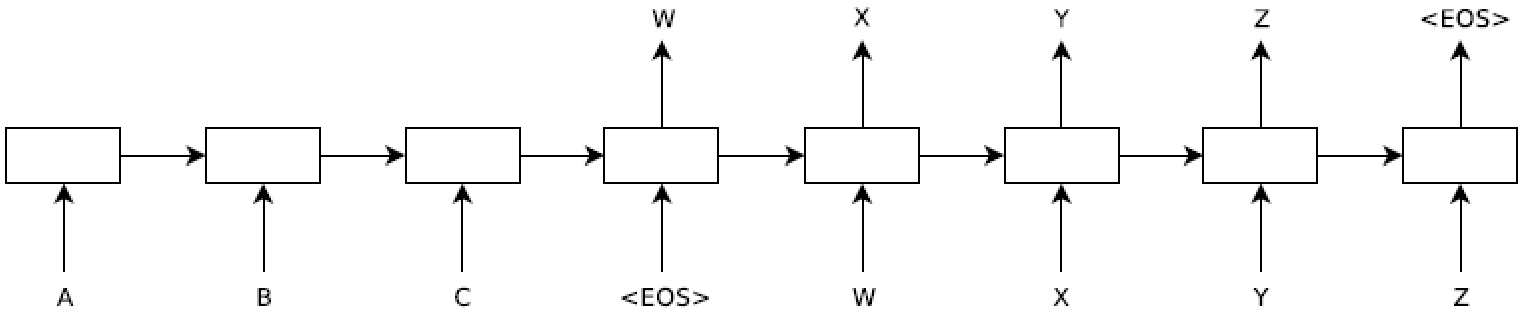
# 百度的Deep speech 2


CTC
Fully Connected
Recurrent or GRU (Bidirectional)
Batch Normalization
1D or 2D Invariant Convolution
Spectrogram

- 可以在超大数据集上训练测试
- 得益于HPC系统
- 正确率从80%到90+%

Amodei et al., 2015

# 句子到句子的翻译



- Input sequence: $x_1, \ldots, x_T$; output sequence: $y_1, \ldots, y_{T'}$
- Estimate: $P(y_1, \ldots, y_{T'} | x_1, \ldots, x_T)$
- Use LSTM to represent input sequence as a vector $v$, then

$$P(y_1, \ldots, y_{T'} | x_1, \ldots, x_T) = \Pi_{t=1}^{T'} P(y_t | v, y_1, \ldots, y_{t-1})$$

| | |
|---|---|
| **Our model** | Ulrich UNK , membre du conseil d' administration du constructeur automobile Audi , affirme qu' il s' agit d' une pratique courante depuis des années pour que les téléphones portables puissent être collectés avant les réunions du conseil d' administration afin qu' ils ne soient pas utilisés comme appareils d' écoute à distance . |
| **Truth** | Ulrich Hackenberg , membre du conseil d' administration du constructeur automobile Audi , déclare que la collecte des téléphones portables avant les réunions du conseil , afin qu' ils ne puissent pas être utilisés comme appareils d' écoute à distance , est une pratique courante depuis des années . |

Sutskever et al., 2014

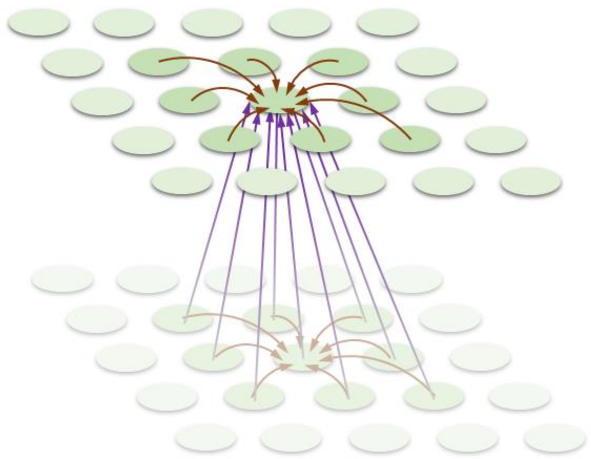
# 前馈和反馈的结合

- Add recurrent connections to CNN
- In the convolutional layer, the net input is

$$z_{ijk}(t) = (\mathbf{w}_k^f)^T \mathbf{u}^{(i,j)}(t) + (\mathbf{w}_k^r)^T \mathbf{x}^{(i,j)}(t-1) + b_k$$

and the output is

$$x_{ijk}(t) = g(f(z_{ijk}(t)))$$

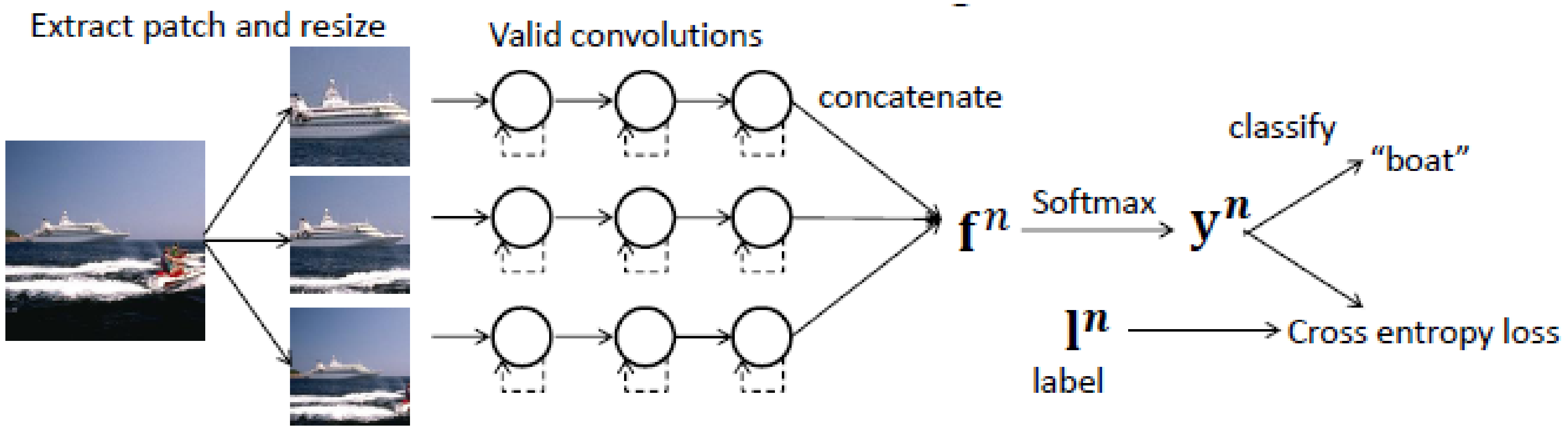


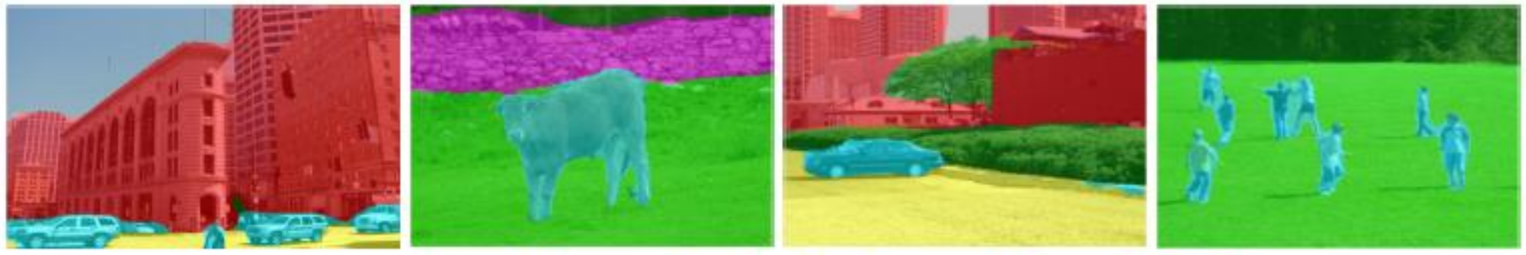
| Model | No. of Param. | Testing Error (%) |
|---|---|---|
| Maxout [17] | > 10 M | 38.57 |
| Prob maxout [47] | > 10 M | 38.14 |
| Tree based priors [49] | — | 36.85 |
| NIN [33] | 0.98 M | 35.68 |
| DSN [30] | 0.98 M | 34.57 |
| RCNN-96 | 0.68 M | **34.18** |
| RCNN-128 | 1.20 M | **32.59** |
| RCNN-160 | 1.87 M | **31.75** |

Table 3. Comparison with existing models on CIFAR-100

Liang, Hu, 2015

# 前馈和反馈的结合

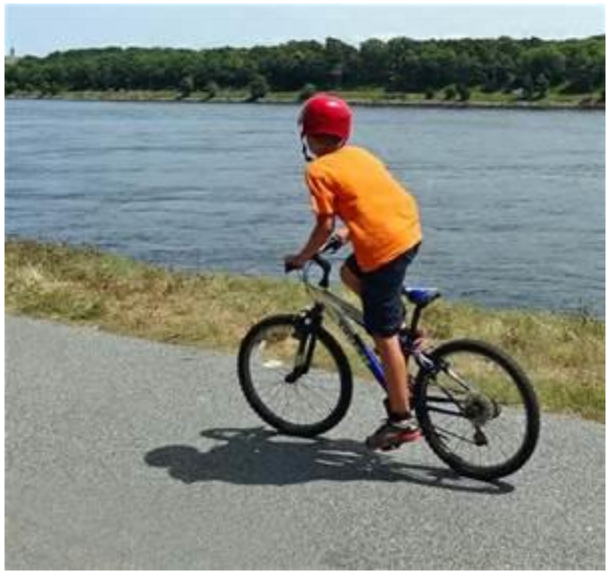




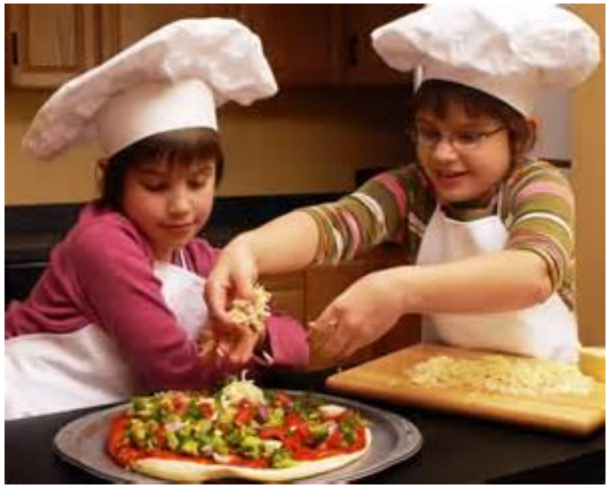
| ⬜ | 🟩 | 🟨 | 🟩 | 🟦 | 🟥 | 🟪 | 🟦 |
|---|---|---|---|---|---|---|---|
| sky | tree | road | grass | water | building | mntn | object |

Liang et al., 2015

## 看图说话

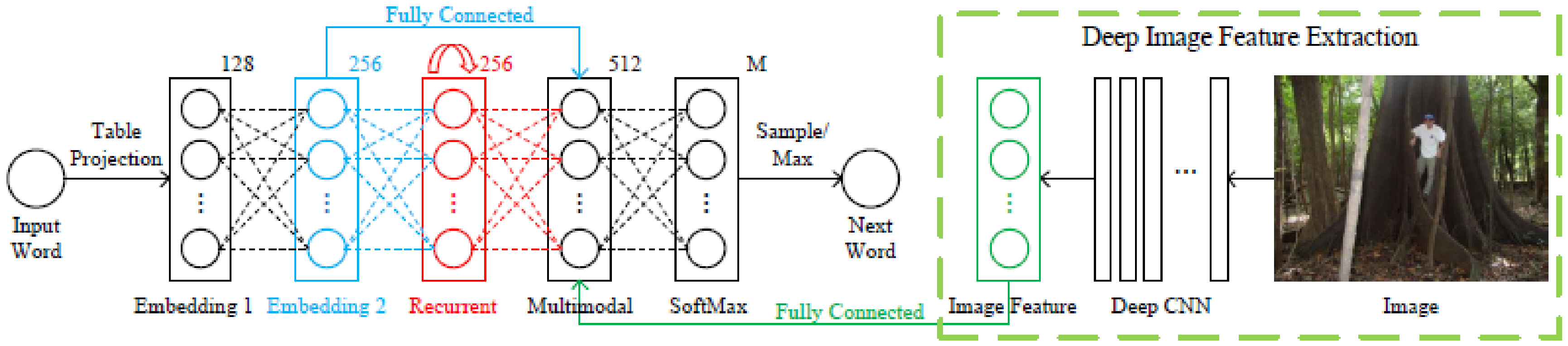


A boy is riding a bike besides a lake.



Two kids are making pizza.

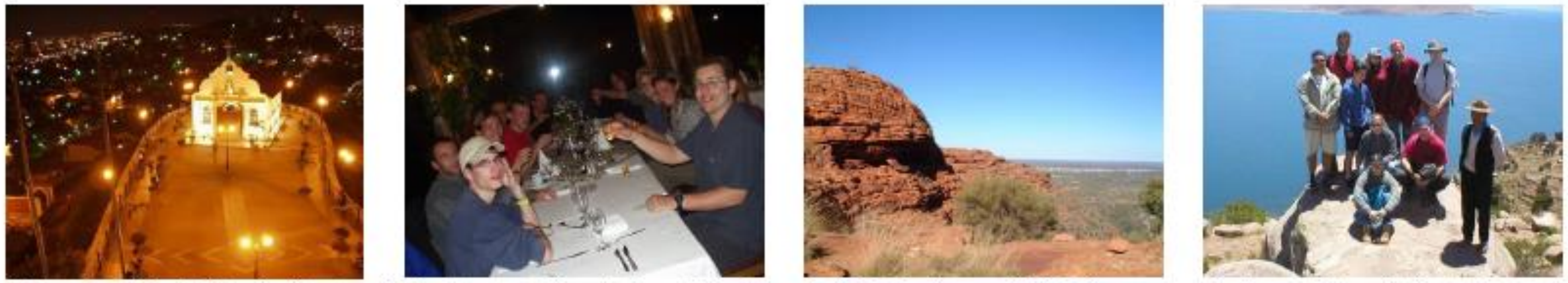
# CNN+RNN进行多模态学习



- One-hot representation for words
- The model will estimate the probability distribution of the next word given previous words and the image
- Word embedding layers are *randomly* initialized
- The 7th-layer features of the AlexNet pretrained on ImageNet are used and *fixed*

Mao et al., 2014

# 产生句子和检索的例子



| | | | | |
|---|---|---|---|---|
| Retr. | 1. Top view of the lights of a city at night, with a well-illuminated square in front of a church in the foreground; 2. People on the stairs in front of an illuminated cathedral with two towers at night; | 1. Tourists are sitting at a long table with beer bottles on it in a rather dark restaurant and are raising their bierglaeser; 2. Tourists are sitting at a long table with a white table-cloth in a somewhat dark restaurant; | 1. A dry landscape with light brown grass and green shrubs and trees in the foreground and large reddish-brown rocks and a blue sky in the background; 2. A few bushes at the bottom and a clear sky in the background; | 1. Group picture of nine tourists and one local on a grey rock with a lake in the background; 2. Five people are standing and four are squatting on a brown rock in the foreground; |
| Gen. | A square with burning street lamps and a street in the foreground; | Tourists are sitting at a long table with a white table cloth and are eating; | A dry landscape with green trees and bushes and light brown grass in the foreground and reddish-brown round rock domes and a blue sky in the background; | A blue sky in the background; |

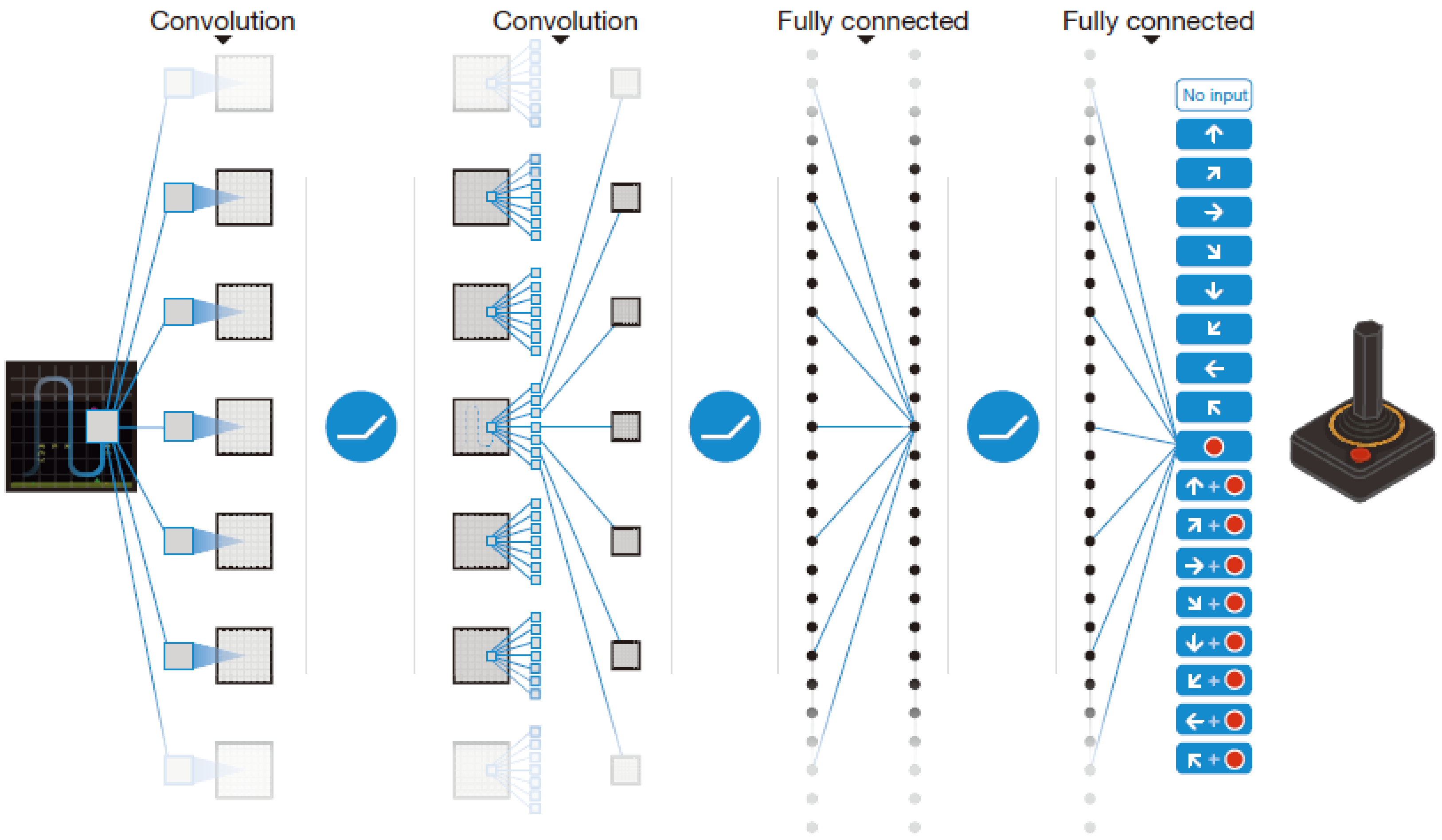
Figure 1: Examples of the generated and two top-ranked retrieved sentences given the query image from IAPR TC-12 dataset. The sentences can well describe the content of the images. We show a failure case in the fourth image, where the model mistakenly treats the lake as the sky.

Mao et al., 2014

# 深度强化学习

- Atari 2600 platform offers 49 games
- Google’s deep Q-network (DQN) performs the same as or better than the human expert in 29 games
- The same network, same learning algorithms

# DQN



Mnih et al., 2015

## Concluding remarks

- Deep learning has achieved exciting results on many real-world problems
- It seems to be a good model for processing big data
- Large models seems to be critical
  - Parallel computing
- Theoretical foundations are lacked
- Unsupervised learning deserves further investigation

# Online resource

- Website: http://deeplearning.net/
  - A reading list
  - Software
  - Datasets
  - Tutorials and demos
- Coding tools
  - Cuda-convnet by Alex Krizhevsky
  - Theano @ University of Montreal
  - Caffe @ UC Berkley
  - Tensor flow by Google
  - Torch by Facebook
  - Deeplearning4j