

# データサイエンス

## 第13回

~ニューラルネットワーク・深層学習~

情報理工学系研究科  
創造情報学専攻  
中山 英樹

# 本日の内容

- 前回の補足・課題中間結果
- ニューラルネットワーク
  - 多層パーセプトロン
  - 深層学習
- 本講義のまとめ

# カーネル法

- 線型モデル  $\hat{f}(\mathbf{x}) = \mathbf{a}^T \mathbf{x}$

- カーネルモデル  $\hat{f}(\mathbf{x}) = \sum_{i=1}^n \alpha_i K(\mathbf{x}, \mathbf{x}_i)$

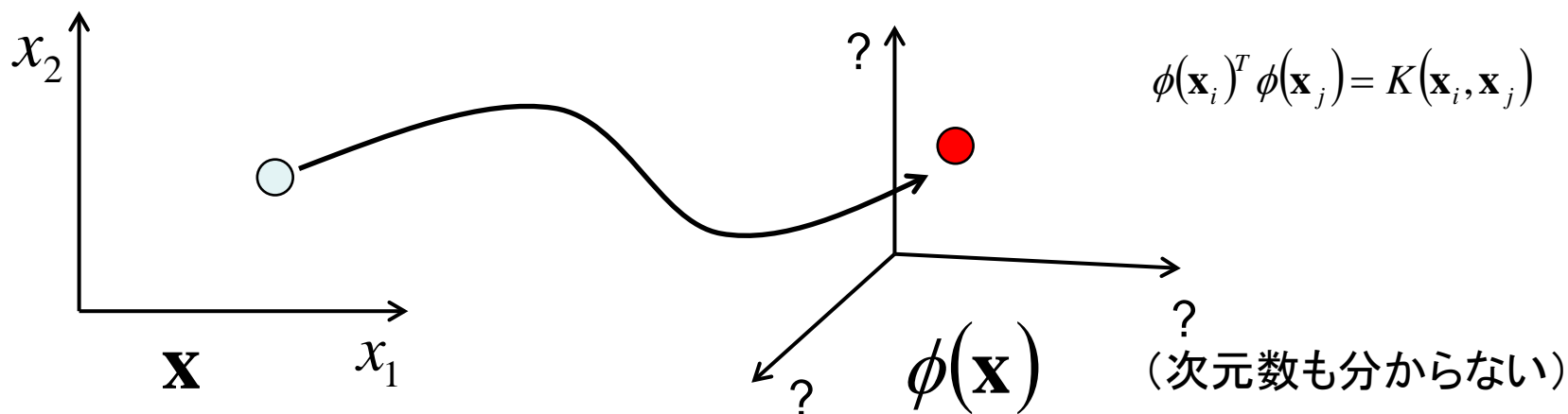
$K(\mathbf{x}_i, \mathbf{x}_j)$  : カーネル関数。サンプル  $\mathbf{x}_i$  と  $\mathbf{x}_j$  の類似度を定義する

(例) ガウスカーネル  $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2c^2}\right)$

- 直感的には、カーネル関数で定義される各学習サンプルとの類似度を新しい特徴とした線形モデル

# カーネルトリック

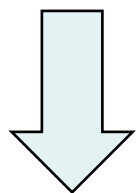
- カーネルモデルは、特徴ベクトル  $\mathbf{x}$  を、内積がカーネル関数  $K$  で定義される高次元空間  $\phi(\mathbf{x})$  に陰に射影する



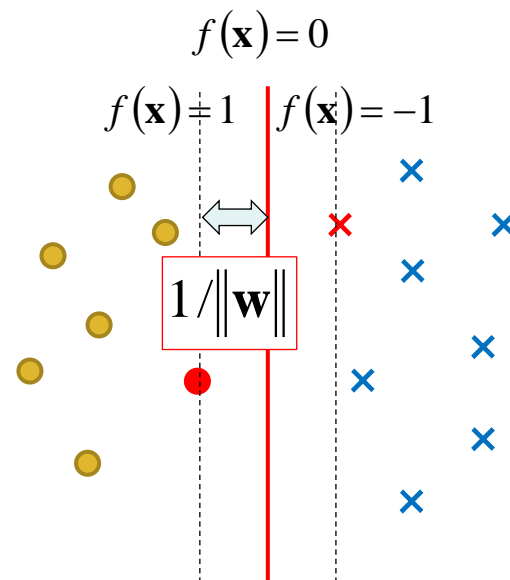
- 実際に  $\phi(\mathbf{x})$  がどのような形かは分からない
  - 多くの線形手法の学習・識別アルゴリズムはカーネルモデルを用いると内積計算だけで実現するように書き換えることができるので、分からなくてもよい =カーネルトリック

# 線形SVM：双対問題

$$\begin{aligned} \min & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{subject to} & y_i f(\mathbf{x}_i) \geq 1 \quad \text{for } \forall i \end{aligned}$$



双対形式に書き換えると…  
(導出は省略)



$$\max \left[ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \underline{\mathbf{x}_i^T \mathbf{x}_j} \right]$$

$$\text{subject to } \underline{\alpha_i \geq 0} \quad \text{for } \forall i$$

$$\sum_{i=1}^n \alpha_i y_i = 0$$

内積しか出てこない

$\alpha_i > 0$  に対応する  $\mathbf{x}_i$  が  
サポートベクター

# カーネルSVM

$$\max \left[ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \underbrace{K(\mathbf{x}_i, \mathbf{x}_j)}_{\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)} \right]$$

subject to  $\alpha_i \geq 0$  for  $\forall i$

$$\sum_{i=1}^n \alpha_i y_i = 0$$

高次元空間における内積を  
元の特徴空間で計算


# カーネル多変量解析

- カーネルPCA

高次元空間 $\phi(\mathbf{x})$ における分散最大化

$$\max_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^N \left( \mathbf{w}^T \left\{ \phi(\mathbf{x}_i) - \frac{1}{N} \sum_{j=1}^N \phi(\mathbf{x}_j) \right\} \right)^2$$

$\text{Var}(\mathbf{w}^T \phi(\mathbf{x}))$




ここで、高次元空間における射影  $\mathbf{w}$  は

サンプル結合  $\mathbf{w} = \sum_{k=1}^N \alpha_k \left\{ \phi(\mathbf{x}_k) - \frac{1}{N} \sum_{l=1}^N \phi(\mathbf{x}_l) \right\}$

を考えれば十分（ここがポイント！）

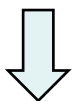
オフセット（サンプル平均）



# カーネルPCA (つづき)

代入して問題を書きかえると

$$\max_{\boldsymbol{\alpha}} \frac{1}{N} \sum_{i=1}^N \left( \sum_{k=1}^N \alpha_k \left\{ \phi(\mathbf{x}_k) - \frac{1}{N} \sum_{l=1}^N \phi(\mathbf{x}_l) \right\} \cdot \left\{ \phi(\mathbf{x}_i) - \frac{1}{N} \sum_{j=1}^N \phi(\mathbf{x}_j) \right\} \right)^2$$

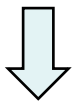


$$\max_{\boldsymbol{\alpha}} \frac{1}{N} \boldsymbol{\alpha}^T \tilde{K}^2 \boldsymbol{\alpha}$$

subject to

$$\boldsymbol{\alpha}^T \tilde{K} \boldsymbol{\alpha} = 1$$

(ノルム1の条件↑)



$$\tilde{K} \boldsymbol{\alpha} = \lambda \boldsymbol{\alpha} \quad (\boldsymbol{\alpha}^T \tilde{K} \boldsymbol{\alpha} = 1) \quad \text{サンプル数次元の固有値問題}$$

$\tilde{K}$  : (中心化) **グラム行列**

$$\tilde{K}_{ij} = \underline{K(\mathbf{x}_i, \mathbf{x}_j)} - \frac{1}{N} \sum_{s=1}^N K(\mathbf{x}_i, \mathbf{x}_s) - \frac{1}{N} \sum_{t=1}^N K(\mathbf{x}_t, \mathbf{x}_j) + \frac{1}{N^2} \sum_{s,t=1}^N K(\mathbf{x}_s, \mathbf{x}_t)$$

要は、カーネルで定義されるサンプル間類似度を要素に持つ行列

# カーネルPCA (つづき)

未知データ点  $\mathbf{x}^q$  のカーネル主成分空間への射影は

$$\begin{aligned}
 & \mathbf{w}^T \left\{ \phi(\mathbf{x}^q) - \frac{1}{N} \sum_{i=1}^N \phi(\mathbf{x}_i) \right\} \\
 &= \left( \sum_{k=1}^N \alpha_k^{(p)} \left\{ \phi(\mathbf{x}_k) - \frac{1}{N} \sum_{l=1}^N \phi(\mathbf{x}_l) \right\} \right) \left\{ \phi(\mathbf{x}^q) - \frac{1}{N} \sum_{i=1}^N \phi(\mathbf{x}_i) \right\} \\
 &= \boldsymbol{\alpha}^{(p)T} \tilde{\mathbf{K}}^s \quad \swarrow \text{p番目の固有ベクトル}
 \end{aligned}$$

$$\tilde{K}_i^s = \underline{K(\mathbf{x}^s, \mathbf{x}_i)} - \frac{1}{N} \sum_{s=1}^N K(\mathbf{x}^q, \mathbf{x}_s) - \frac{1}{N} \sum_{t=1}^N K(\mathbf{x}_i, \mathbf{x}_t) + \frac{1}{N^2} \sum_{s,t=1}^N K(\mathbf{x}_s, \mathbf{x}_t)$$

- 多変量解析手法は一般に同様の枠組でカーネル化できる
  - 回帰分析、判別分析、正準相関分析

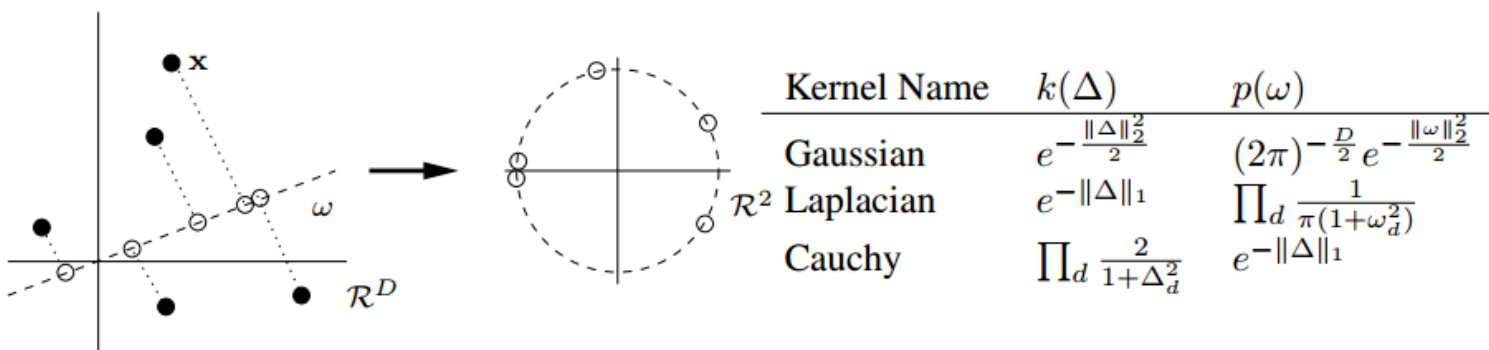
# Explicit feature maps

- 高次元空間  $\phi(\mathbf{x})$  を陽に導出するアプローチ
  - 多くの場合は近似的
- もちろんいつでも都合よく求まるわけではない
  - 実用上よく使われるものを中心に研究されている
  - ヒストグラム向けのカーネルなど
- データの大規模化に伴い、計算コストのボトルネックが逆転
  - 特徴次元数を大幅に増やしてでも、サンプル数に対して線形の計算コストに収めたい

# より一般的な場合

- RBF kernel (ガウスカーネル)

- A. Rahimi and B. Recht, "Random features for large-scale kernel machines", In Proc. NIPS 2007.
- Random Fourier features による近似



- Generalized RBF kernel

- S. Vempati, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "Generalized RBF feature maps for Efficient Detection", In Proc. BMVC 2010.

RBF kernel + additive kernel (例えばこんなの)

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{2\sigma^2} \chi^2(\mathbf{x}, \mathbf{y})\right), \quad \chi^2(\mathbf{x}, \mathbf{y}) = \frac{1}{2} \sum_{m=1}^d \frac{(x_m - y_m)^2}{x_m + y_m}$$

# Feature Mapsが導出できない場合

- カーネルPCAでユークリッド空間へ埋め込む
  - カーネル主成分を新たな特徴として用いる
- カーネル化の基底に用いるサンプルは少数を選ぶ
  - 選び方はいろいろ研究されている  
(Nyström's approximation)
  - あるいは、単にランダムに選ぶ

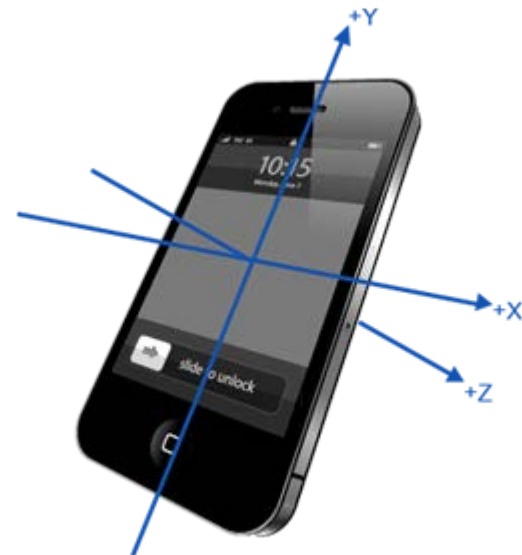
F. Perronnin, J. Sánchez, and Y. Liu. “Large-scale image categorization with explicit data embedding”. In Proc. CVPR, 2010.

# 第二回レポート課題（クラス分類）

- Human Activity Recognition Using Smartphones Sensor Data

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine. International Workshop of Ambient Assisted Living (IWAAL 2012). Vitoria-Gasteiz, Spain. Dec 2012.

- スマートフォンの慣性センサデータ等からユーザの行動を推定
- 30人の被験者データ
- 特徴量（説明変数）：561種類
- 行動カテゴリ（目的変数）：6種類
  - 1. WALKING
  - 2. WALKING UPSTAIRS
  - 3. WALKING DOWNSTAIRS
  - 4. SITTING
  - 5. STANDING
  - 6. LAYING



<http://classmethod.s3.amazonaws.com/wp-content/uploads/2012/06/gyrocompass.png>

# 中間結果

## バリデーションセット

Rank	Name	Accuracy
1	KeisukeMaeda	0.91384615384615
2	KentaroNishi	0.91153846153846
3	chika	0.90461538461538
4	ChihiroW	0.90153846153846
5	DeepLearner	0.90076923076923
6	YusukeFuruta	0.89846153846154
7	TatsuyaKurokawa	0.89538461538462
8	mhidaka	0.89461538461538
8	ChieKamada	0.89461538461538
10	nakachanOK	0.88
11	hatamu	0.87769230769231
12	MutsukiKojima	0.87230769230769
13	ymori	0.86230769230769
14	baseline_svm	0.86153846153846
15	tf	0.85923076923077

## テストセット

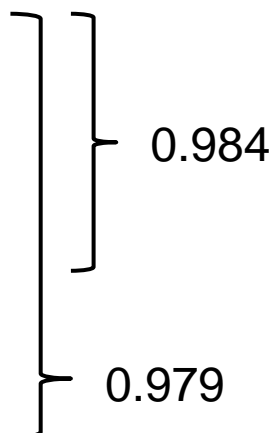
Rank	Name	Accuracy
1	KeisukeMaeda	0.98141529664046
1	KentaroNishi	0.98141529664046
3	YusukeFuruta	0.9807005003574
4	ChihiroW	0.97927090779128
5	ChieKamada	0.97569692637598
6	TatsuyaKurokawa	0.96997855611151
7	chika	0.96926375982845
8	mhidaka	0.96854896354539
9	ymori	0.96783416726233
9	baseline_svm	0.96783416726233
11	tf	0.96568977841315
12	hatamu	0.95067905646891
13	DeepLearner	0.94496068620443
14	nakachanOK	0.94353109363831
15	MutsukiKojima	0.94210150107219

# アンサンブル

- 多数決の結果

## Score Board (on the testing set)

Rank	Name	Accuracy
1	KeisukeMaeda	0.98141529664046
1	KentaroNishi	0.98141529664046
3	YusukeFuruta	0.9807005003574
4	ChihiroW	0.97927090779128
5	ChieKamada	0.97569692637598
6	TatsuyakKurokawa	0.96997855611151
7	chika	0.96926375982845
8	mhidaka	0.96854896354539
9	ymori	0.96783416726233



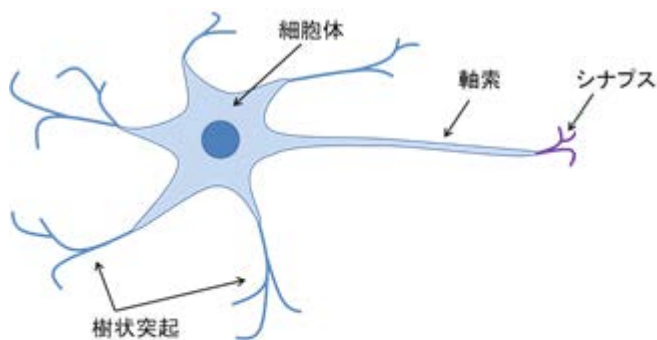
# 締め切り

- レポート提出締め切り 2月15日（土）
- コンペについて
  - 2月11日（火） 18:00にもう一度ランキング（最終）

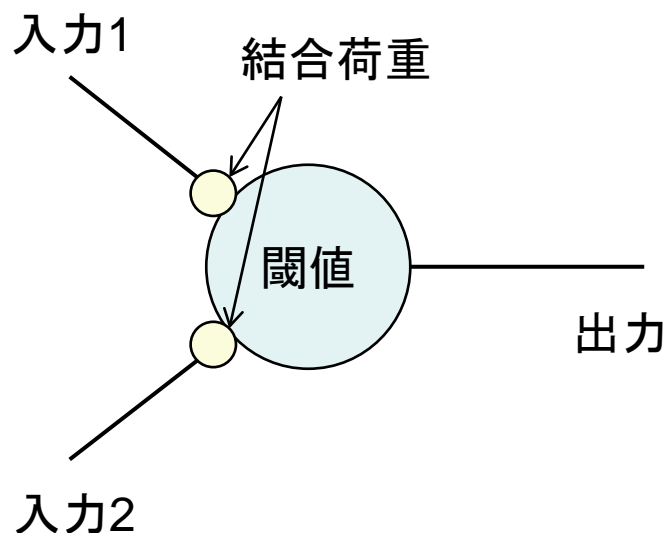
# (人工) ニューラルネットワーク

- 脳神経系を模した数学モデル
- ネットワークを形成する多数の人工ニューロンのシナプス結合強度を変化させて問題解決能力を獲得する

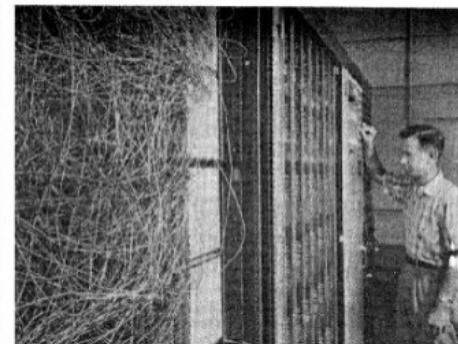
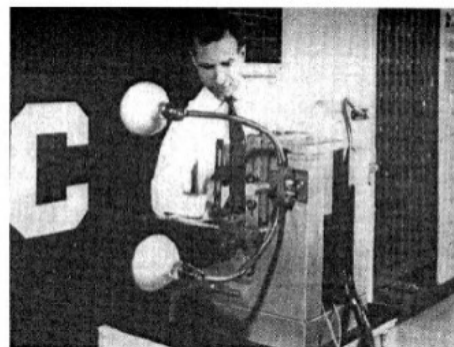
神経細胞（ニューロン）



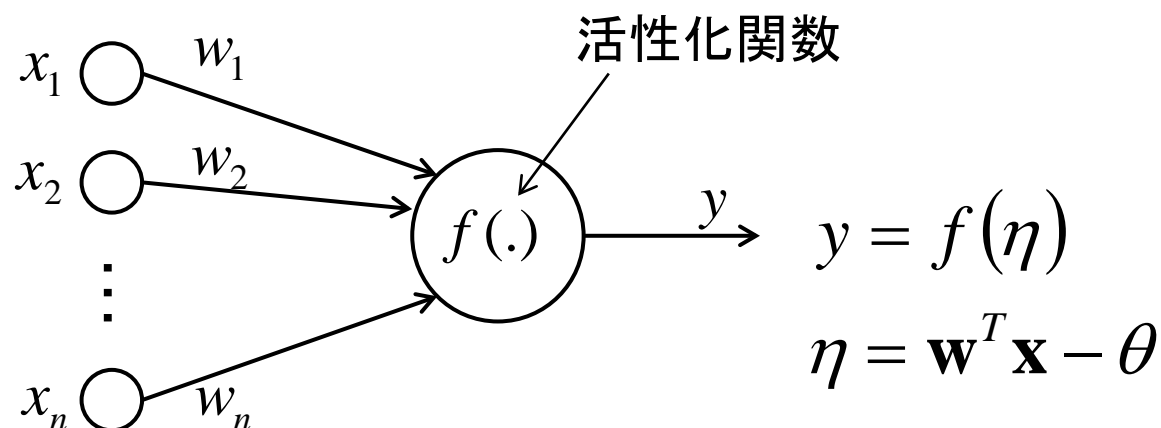
ニューロンモデル



# ニューロンモデル



- 単純パーセプトロン



$$f(\eta) = \begin{cases} 1 & \eta > 0 \\ 0 & \text{otherwise} \end{cases}$$

ステップ関数  
McCulloch & Pitts モデル (1943)

$$f(\eta) = \eta \rightarrow \text{線形回帰と等価}$$

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

シグモイド関数: 古典的かつ最もスタンダード  
(ロジスティック回帰と等価)

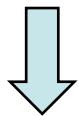
# 単純パーセプトロンの学習

- 活性化関数がシグモイド関数の場合

訓練データ集合  $\{\mathbf{x}_i, y_i\}_{i=1}^N, y_i \in \{0, 1\}$

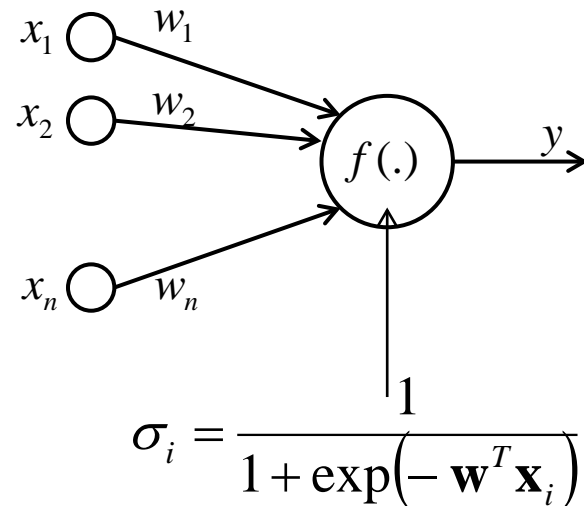
尤度関数は 
$$L = \prod_{i=1}^N \sigma_i^{y_i} \{1 - \sigma_i\}^{1-y_i}$$

負の対数尤度は 
$$E(\mathbf{w}) = -\ln L = \sum_{i=1}^N \{y_i \ln \sigma_i + (1 - y_i) \ln(1 - \sigma_i)\}$$



$$\frac{E(\mathbf{w})}{\partial \mathbf{w}} = \sum_{i=1}^N (\sigma_i - y_i) \mathbf{x}_i$$

エラーに説明変数をかけたもの

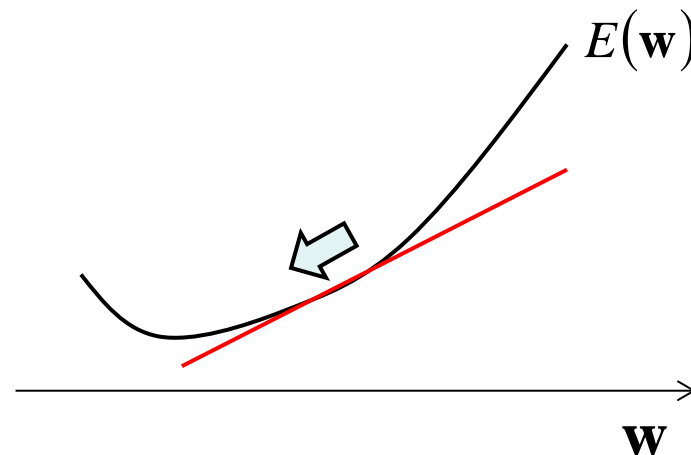


# 最急降下法

- 収束するまで少しずつ勾配方向へ引っ張る
  - $\alpha$  は微小な正のパラメータ

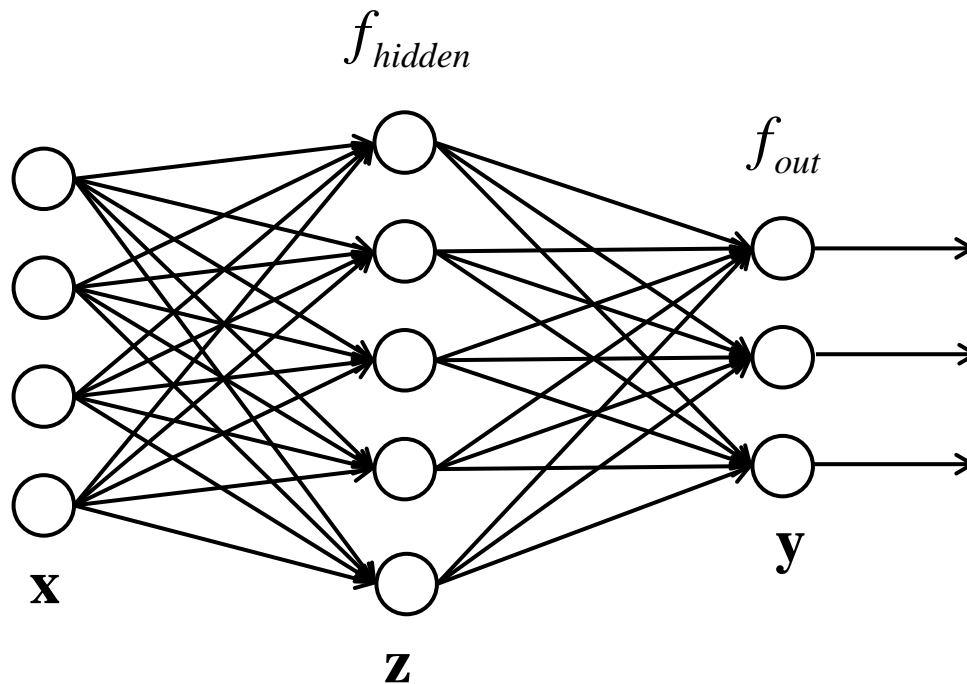
$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \sum_{i=1}^N (\sigma_i - y_i) \mathbf{x}_i$$

$\frac{E(\mathbf{w})}{\partial \mathbf{w}}$



# 多層パーセプトロン

- 十分な数の素子があれば、任意の連続関数は 3 層のニューラルネットワークで近似できる
  - 誤差逆伝播法とでパラメータを最適化



# 誤差逆伝播法

評価関数（二乗誤差）：

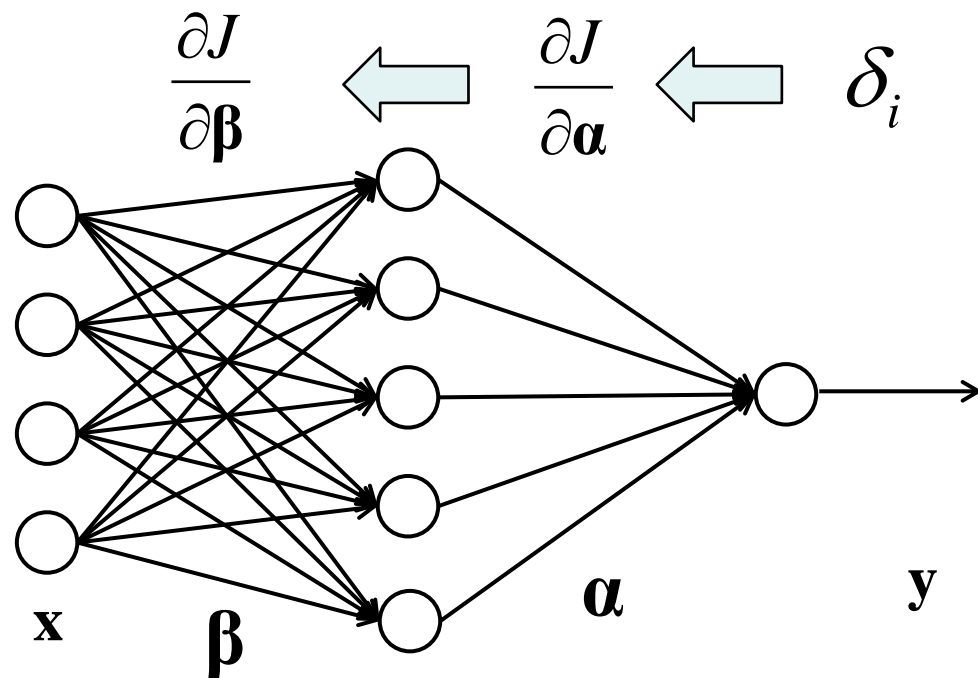
$$J = \sum_{i=1}^N (f(\mathbf{x}_i) - y_i)^2$$

活性化関数は全て  
シグモイド関数とすると

$$\frac{\partial J}{\partial \alpha_j} = 2 \sum_{i=1}^N \sigma_{i,j} \delta_i \quad \leftarrow \delta_i = f(\mathbf{x}_i) - y_i$$

$$\frac{\partial J}{\partial \beta_j^{(k)}} = 2 \alpha_j \sum_{i=1}^N \sigma_{i,j} (1 - \sigma_{i,j}) \delta_i x_i^{(k)}$$

$$\sigma_{i,j} = \frac{1}{1 + \exp(-\boldsymbol{\beta}_j^T \mathbf{x}_i)}$$



発生した誤差に対し、前層での出力が強いニューロンが大きく更新される

# 古典的多層ニューラルネットの問題点

- 誤差逆伝播学習は実際にはあまりうまくいかず…
  - 初期値依存性が強い（局所最適解）
  - 過学習しやすい
  - 入力に近い層の学習が遅い（誤差が拡散されてしまう）

# ニューラルネット冬の時代

- とある論文の冒頭 (2003年)

After being extremely popular in the early 1990s, **neural networks have fallen out of favor in research in the last 5 years.**

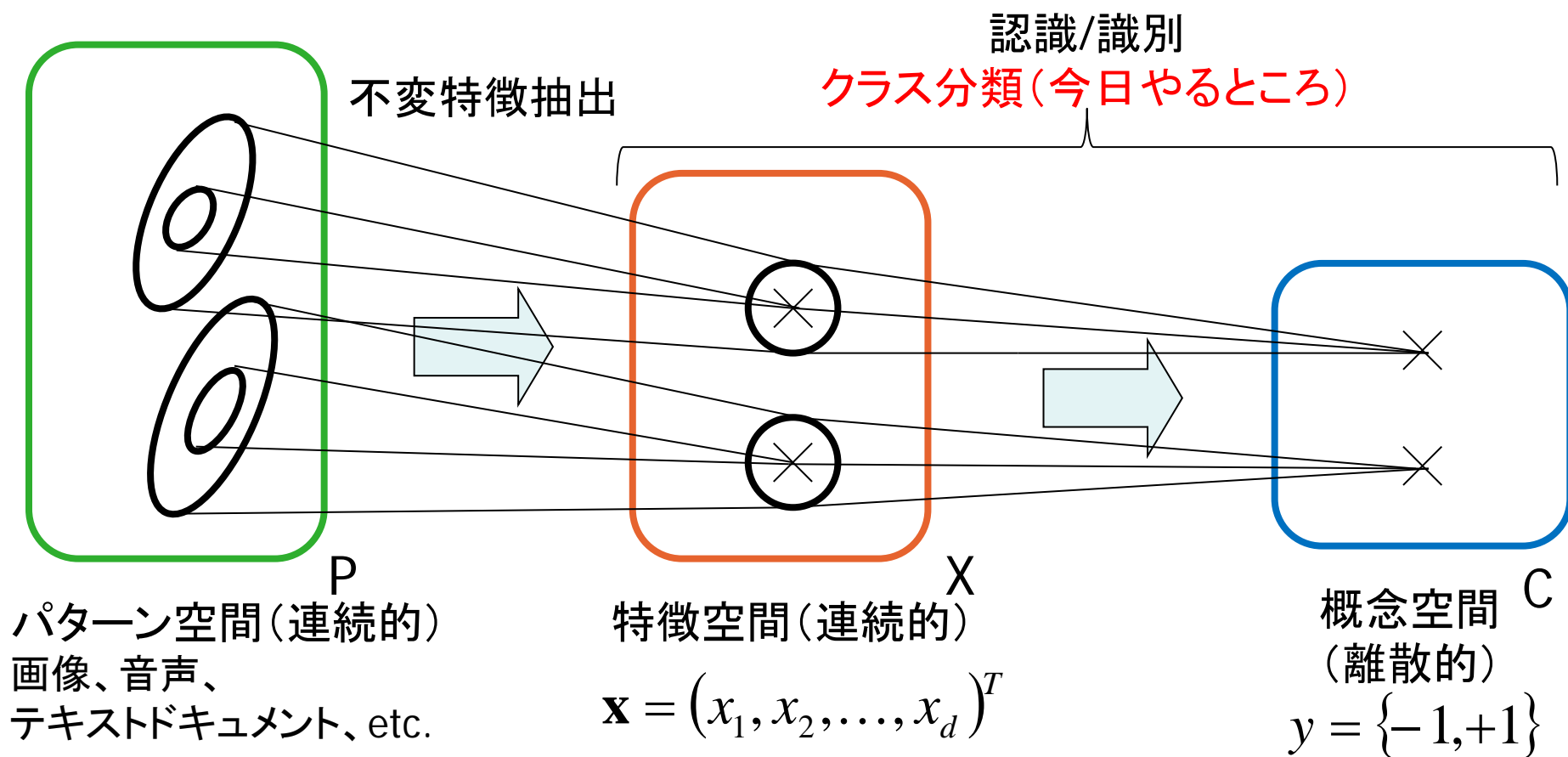
In 2000, it was even pointed out by the organizers of the Neural Information Processing System (NIPS) conference that the term **“neural networks” in the submission title was negatively correlated with acceptance.** In contrast, positive correlations were made with support vector machines (SVMs), Bayesian networks, and variational methods.  
[Simard et al., ICDAR'03]

# Deep learning (深層学習)

- 従来扱われてきたよりもさらに多層のニューラルネット
  - 現在は7~8層くらいが多い？
- 生データから目的変数に至る深い構造を通して学習
  - 特徴量（に相当する構造）も自動的に獲得
  - パターン認識の文脈では表現学習（representation learning）とほぼ同義で扱われることも
- 音声認識・画像認識・自然言語処理などさまざまな分野で圧倒的な性能を達成

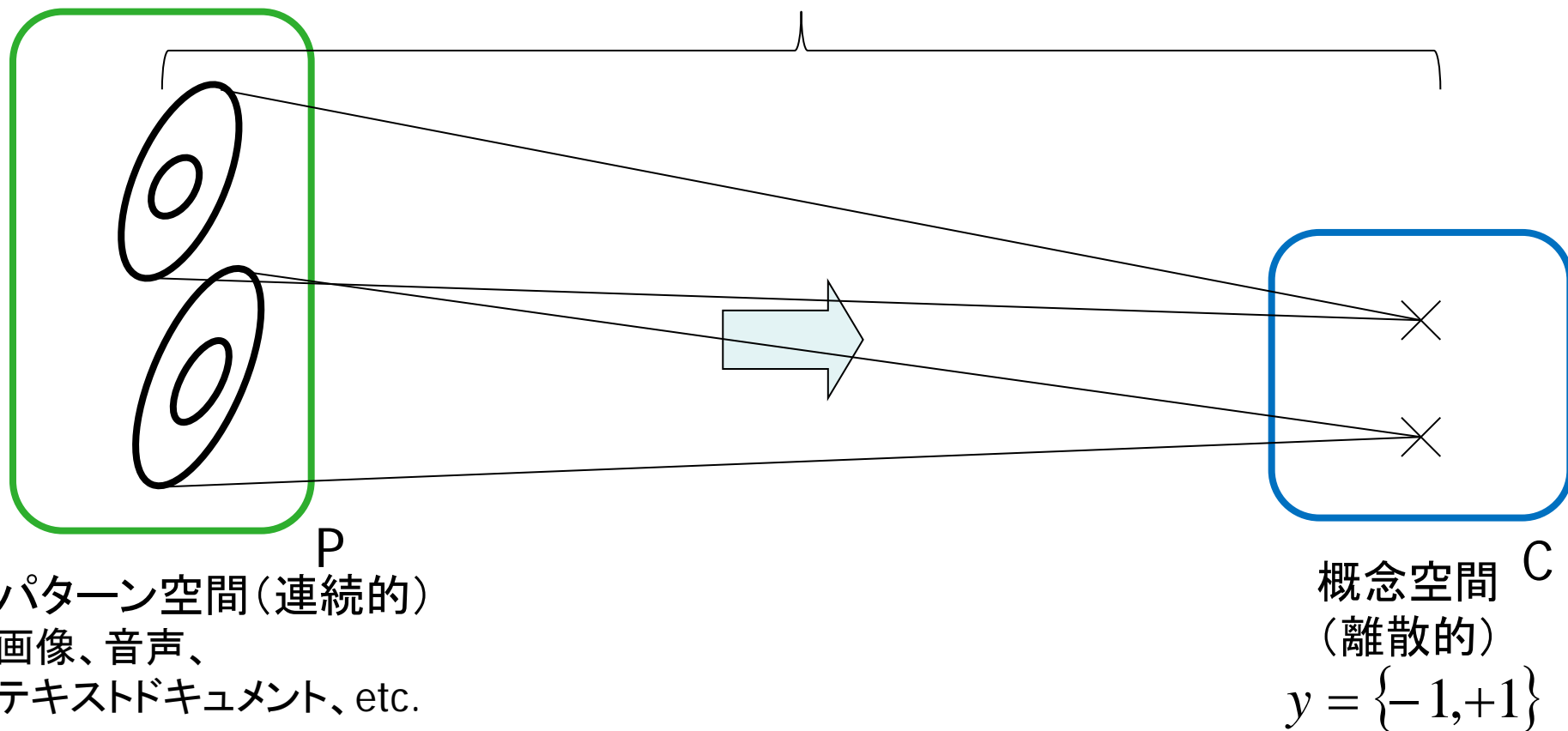
# パターン認識のパイプライン

- よい特徴量（説明変数）を抽出・選択することは極めて重要
  - Garbage in garbage out

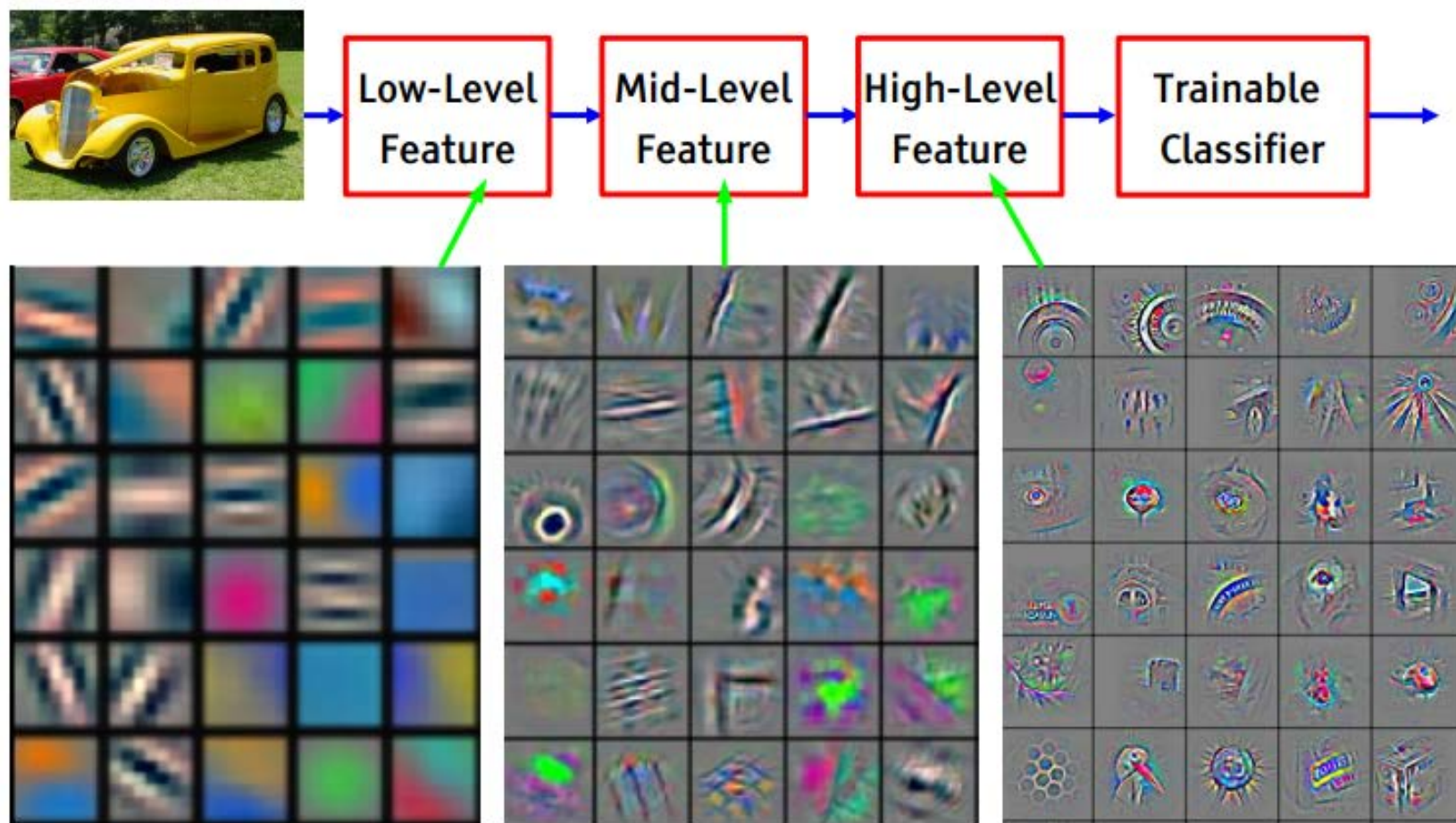


# こうなりつつある

入出力の全構造を学習



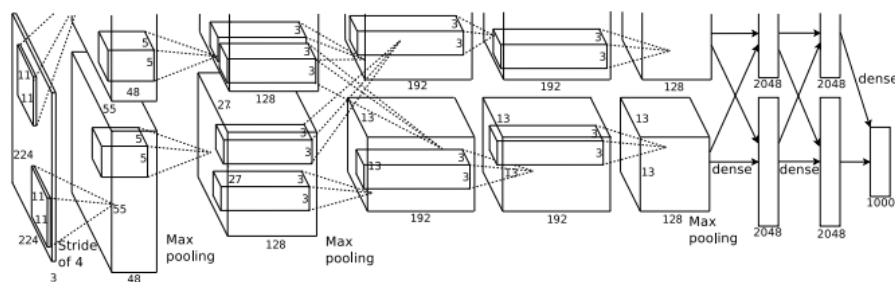
# 画像認識の例



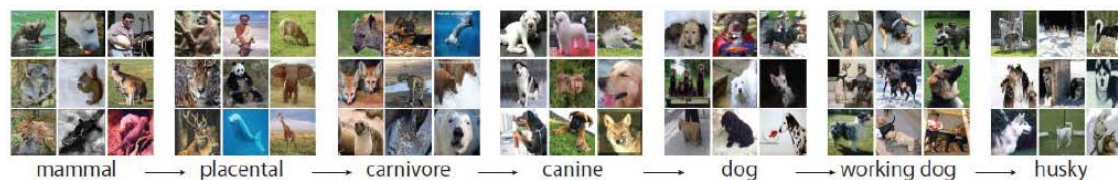
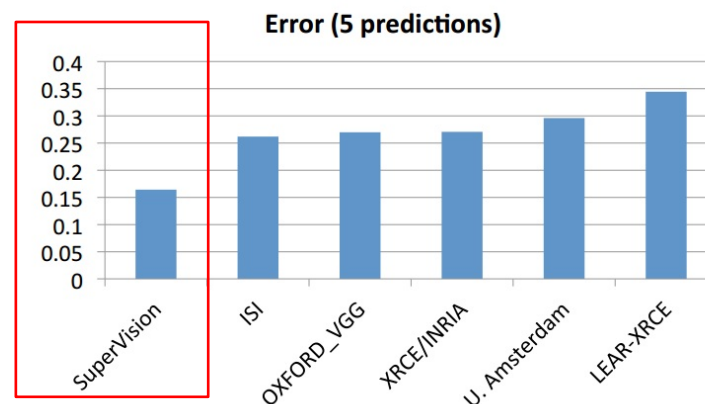
<http://www.cs.nyu.edu/~yann/talks/lecun-ranzato-icml2013.pdf>

# 画像認識の例

- ImageNet large-scale visual recognition challenge 2012
  - 1000カテゴリの画像認識コンペティション



[Krizhevsky et al. 2012]



- 他、文字認識、道路標識認識などの主要なデータセットで人間に匹敵する識別精度を達成

## Audio

TIMIT Phone classification	Accuracy
Prior art (Clarkson et al., 1999)	79.6%
Stanford Feature learning	<b>80.3%</b>

TIMIT Speaker identification	Accuracy
Prior art (Reynolds, 1995)	99.7%
Stanford Feature learning	<b>100.0%</b>

## Images

CIFAR Object classification	Accuracy
Prior art (Krizhevsky, 2010)	78.9%
Stanford Feature learning	<b>81.5%</b>

NORB Object classification	Accuracy
Prior art (Ranzato et al., 2009)	94.4%
Stanford Feature learning	<b>97.3%</b>

## Video

Hollywood2 Classification	Accuracy
Prior art (Laptev et al., 2004)	48%
Stanford Feature learning	<b>53%</b>
KTH	Accuracy
Prior art (Wang et al., 2010)	92.1%
Stanford Feature learning	<b>93.9%</b>

YouTube	Accuracy
Prior art (Liu et al., 2009)	71.2%
Stanford Feature learning	<b>75.8%</b>
UCF	Accuracy
Prior art (Wang et al., 2010)	85.6%
Stanford Feature learning	<b>86.5%</b>

## Multimodal (audio/video)

AVLetters Lip reading	Accuracy
Prior art (Zhao et al., 2009)	58.9%
Stanford Feature learning	<b>65.8%</b>

Other unsupervised feature learning records:  
Pedestrian detection (Yann LeCun)  
Different phone recognition task (Geoff Hinton)  
PASCAL VOC object classification (Kai Yu)

# 周辺状況

- Google

- DNNresearch (ILSVRC'12 winnerの会社) 買収 (G. Hinton)
- Google brain project (A. Ng)
- DeepMind 買収 (4億ドル!)

AI研究者Yoshua Bengio氏の見方によると、世界にディープラーニングの優れた専門家が50人くらいしかいないという。でも以下の特許例からも推測できるように、DeepMindが優れたディープラーニング研究者を擁していることが、AIコミュニティ内では知られていた。DeepMindを買収すれば、世界のトップ研究者50人のうち12人を一気に抱えることができるのだ。

- Facebook

- AI Lab 設立 (2013)  
Y. LeCun (所長), M. Ranzato, R. Fergus

[http://www.huffingtonpost.jp/zenichiro-tanaka/google-manhattan-plan\\_b\\_4701358.html](http://www.huffingtonpost.jp/zenichiro-tanaka/google-manhattan-plan_b_4701358.html)

- 音声認識では既に実用化
- 画像認識も？ (Google Picasa)

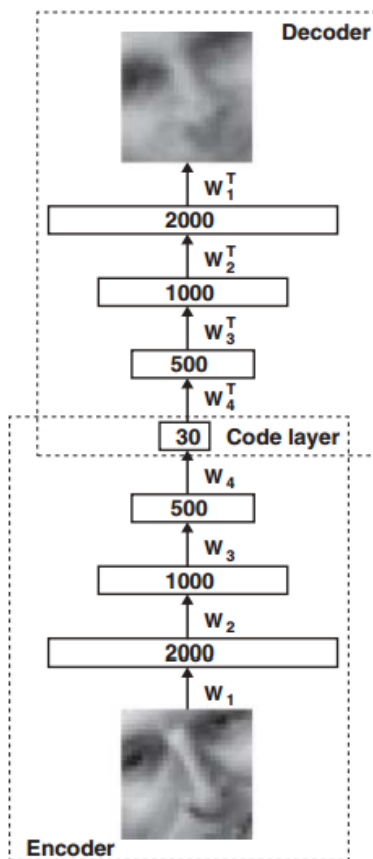
# 昔と何が変わったのか？

- 0. データの増加、計算機の高速化
- 1. ネットワーク初期化手法
- 2. 最適化手法
- 3. 過学習回避の手法
- 4. その他ノウハウの蓄積

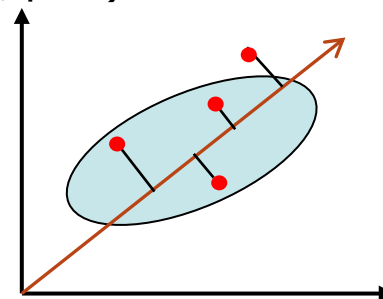
本質的に何かが変わったわけではない（と思う…）

# 火付け役となった研究

- Hinton , G. E. and Salakhutdinov, R. R. “Reducing the dimensionality of data with neural networks”, Science, Vol. 313, No. 5786, pp. 504 - 507, 2006.



- データの圧縮にdeep neural networkを利用
- 元のデータをできるだけ復元できるように圧縮表現を学習
- やっていることはPCAと同じ（ただし非線形、深い）

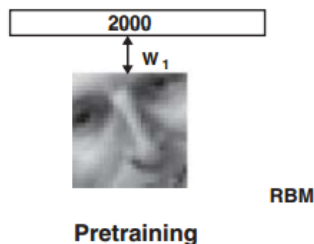


# ポイント

- 全層通しでいきなり学習すると  
うまくいかない！

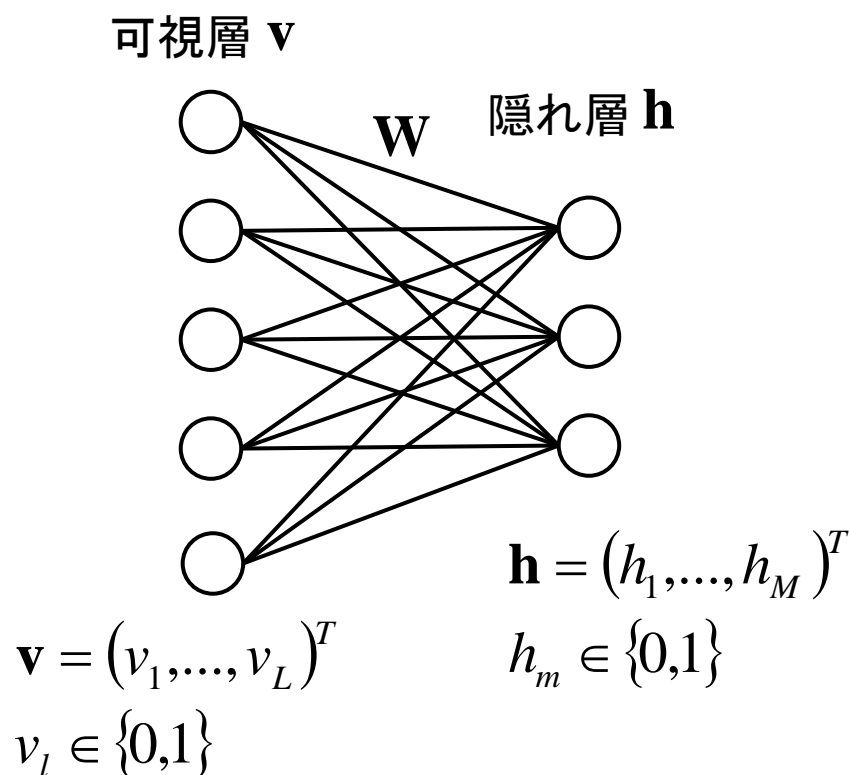


- 一層ごとに独立に、下から順番に  
各層を学習する
  - Greedy layer-wise pre-training
  - これをネットワークの初期状態と  
する
- 初期化後に、通しでネットワーク  
のfine-tuningを行う



# Layer-wise training の手法 (1)

- Restricted Boltzmann machine (RBM)
  - 可視層（入力）と隠れ層（圧縮表現）からなる無向二部グラフ



同時確率分布：

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

エネルギー：

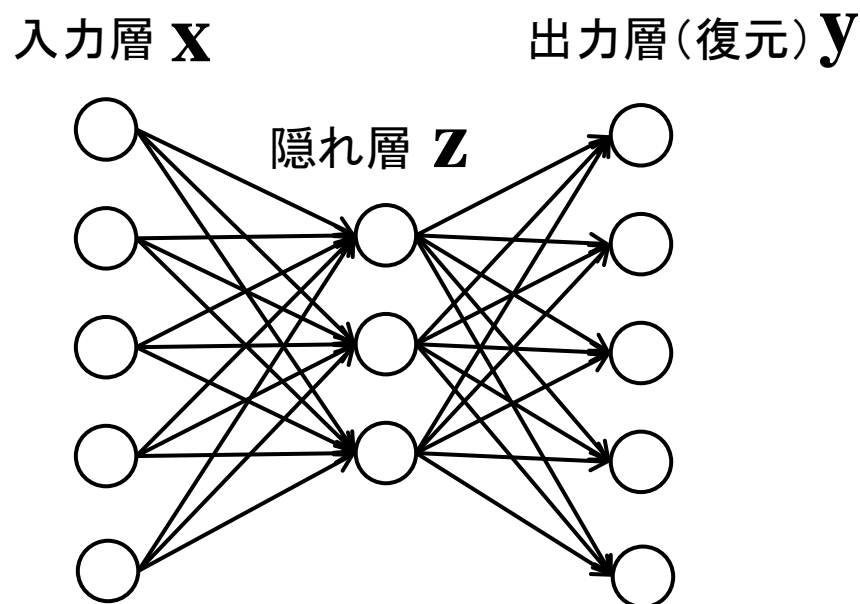
$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^T \mathbf{v} - \mathbf{c}^T \mathbf{h} - \mathbf{v}^T \mathbf{W} \mathbf{h}$$

パラメータ  $\theta = (\mathbf{b}, \mathbf{c}, \mathbf{W})$

尤度  $p(\mathbf{v}) = \sum_{\mathbf{h}} p(\mathbf{v}, \mathbf{h})$   
を最大とするようにパラメータを  
勾配法で学習  
CD法 (Contrastive Divergence)  
による近似を用いる

# Layer-wise training の手法 (2)

- Autoencoder
  - 入力を復元する二層のニューラルネット
  - 隠れ層のニューロン数は入力層より少ない
  - 正則化が重要 (Sparse AE, Denoising AE, etc.)



Encoder:  $\mathbf{z} = f(\mathbf{W}\mathbf{x} + \mathbf{b})$

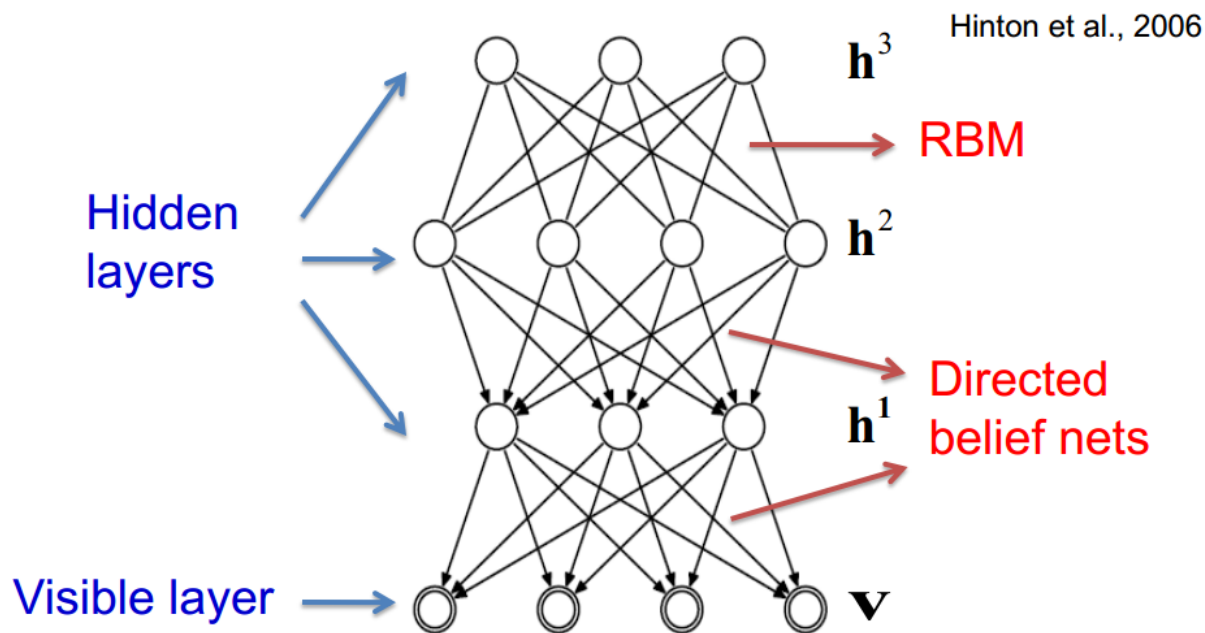
Decoder:  $\mathbf{y} = f(\mathbf{W}'\mathbf{z} + \mathbf{b}')$

$$\mathbf{W}' = \mathbf{W}^T$$

とすることが多い  
(tied weights)

# 教師なし深層ネットワーク (1)

- Deep Belief Network (DBN) [Hinton et al., 2006]
  - 最終層だけ無向のネットワーク
  - 各層をRBMで初期化（可視層から事後確率を伝播）
  - 最後にfine-tuning

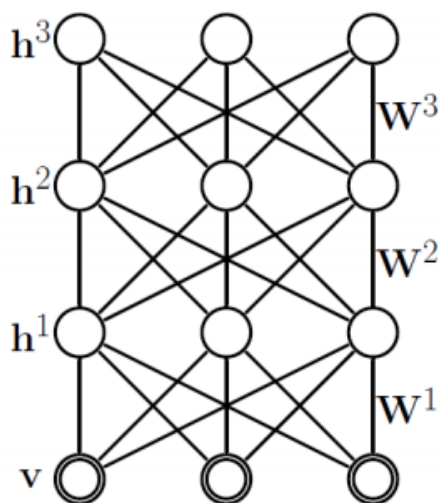


$$P(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2, \dots, \mathbf{h}^l) = P(\mathbf{v} | \mathbf{h}^1)P(\mathbf{h}^1 | \mathbf{h}^2) \dots P(\mathbf{h}^{l-2} | \mathbf{h}^{l-1})P(\mathbf{h}^{l-1}, \mathbf{h}^l)$$

# 教師なし深層ネットワーク (2)

- Deep Boltzmann Machine (DBM) [Salakhutdinov & Hinton, 2009]
  - 全層とも無向のモデル（各層は両側とインタラクション）
  - RBMで各層を事前学習した後、全体をfine-tuning

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



Undirected connections between  
all layers  
(no connections between the  
nodes in the same layer)

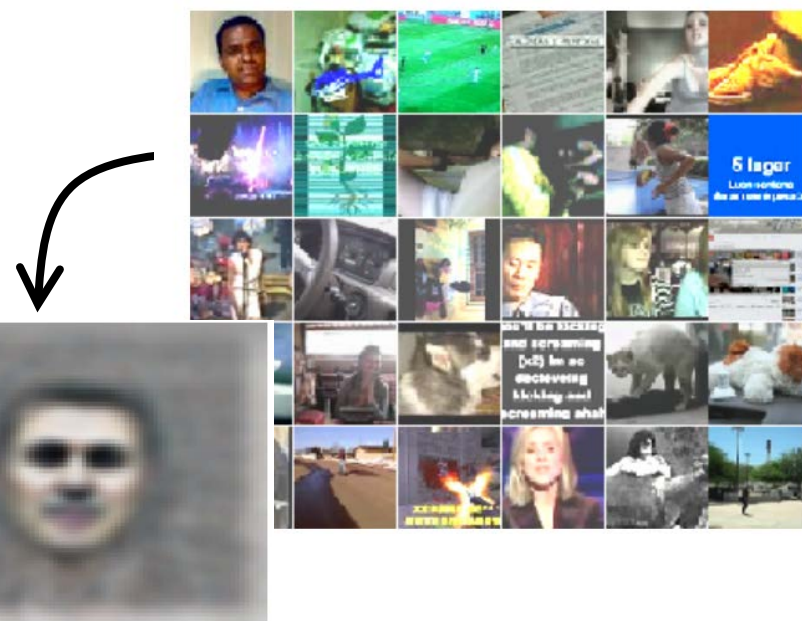
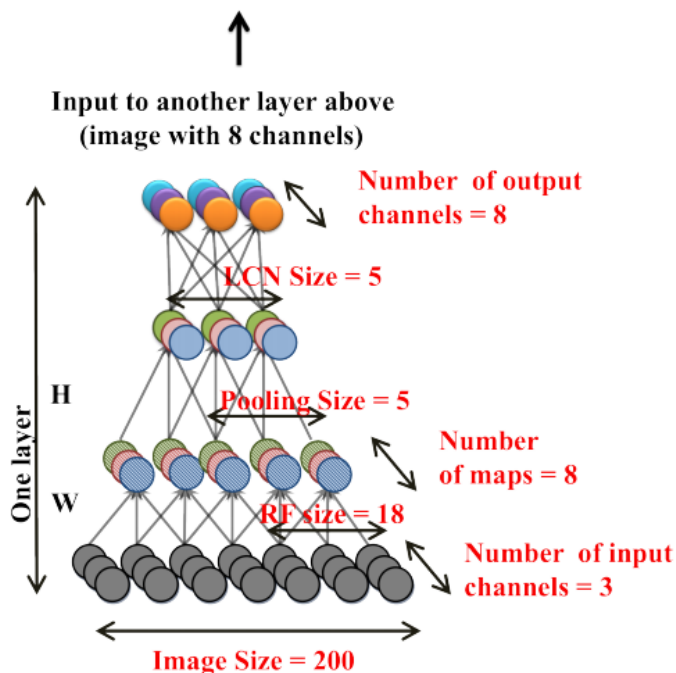
# 教師なし深層ネットワーク(3)

- 他にもいろいろ
  - Stacked (denoising) auto-encoders
  - Stacked predictive sparse coding
  - ...
- 識別タスクに用いる時は
  - 最終層の出力を特徴量として使う（SVM等をかぶせる）
  - さらに誤差逆伝播法を適用してfine-tuning

# 画像特徴の教師なし学習

Le et al., “Building High-level Features Using Large Scale Unsupervised Learning”, ICML’12

- 9-layered locally connected sparse autoencoder
- 1 billion parameters
- A cluster with 1,000 machines (16,000 cores) for three days.
- Fed 10 million YouTube images



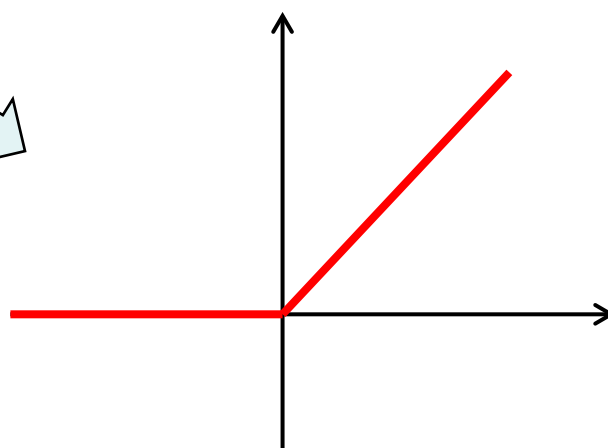
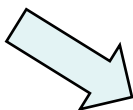
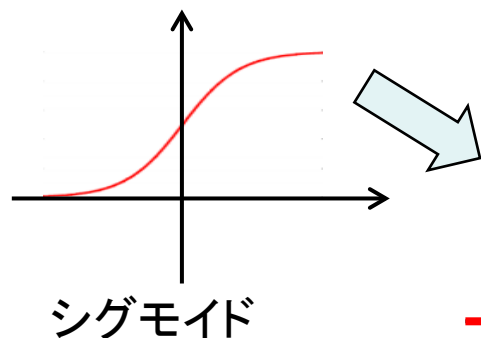
人の顔に特異的に反応するニューロンが自動的に獲得された（他、猫なども）

# 昔と何が変わったのか？

- 0. データの増加、計算機的高速化
- 1. ネットワーク初期化手法
- 2. 最適化手法
  - 活性化関数の工夫
  - 最適化手法の高速化・並列化
- 3. 過学習回避の手法
- 4. その他ノウハウの蓄積

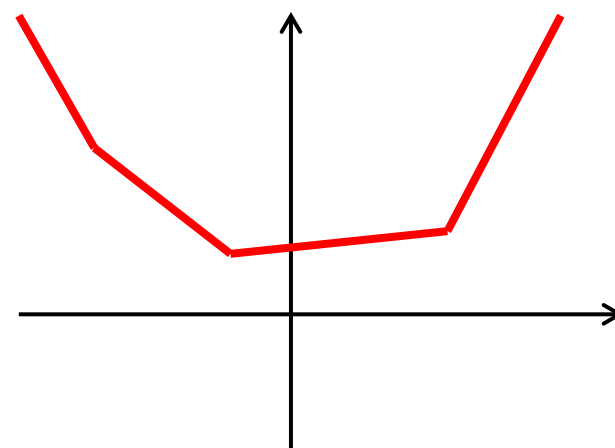
# 活性化関数の工夫

- 勾配が出やすいように関数の設計を工夫
- 区分線形関数が良好な性能を示すことが分かってきた



Rectified linear units (ReLU)  
[Nair & Hinton, 2010]

$$\max(0, x)$$



Maxout [Goodfellow, 2013]

多数の線形関数のmax  
(任意の閾値関数を近似)

c.f. Probabilistic maxout,  
Channel-out...

# 最適化手法自体の発達

- 確率的勾配降下法 (stochastic gradient descent)

※深層学習のために出てきたものではない

- 1サンプルごとに目的関数の勾配を出し、重みを更新
- 学習が圧倒的に高速化  
(注意) 学習サンプルはシャッフルしておくこと

- 例) ロジスティック回帰の場合

- 最急降下法

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \sum_{i=1}^N (\sigma_i - y_i) \mathbf{x}_i$$

- 確率的勾配降下法

$$\mathbf{w} \leftarrow \mathbf{w} - \gamma (\sigma_i - y_i) \mathbf{x}_i$$

# ミニバッチによるSGD

- ある程度サンプルを束ねて更新

$$\mathbf{w} \leftarrow \mathbf{w} - \gamma \frac{1}{B} \sum_{i=1}^B (\sigma_i - y_i) \mathbf{x}_i$$

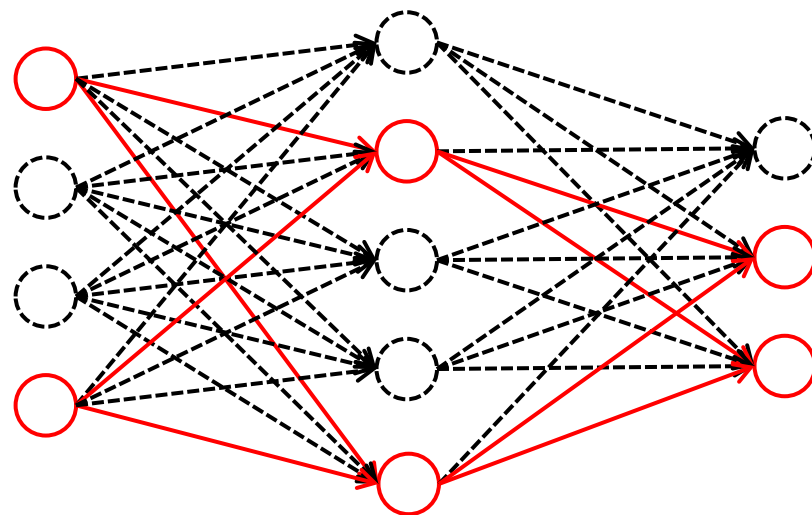
- バッチごとの評価関数の計算は並列化可能
  - 一般にSGDの並列化は難しいが、GPUの実装法まで含めて研究が進められている  
Coates et al., “Deep learning with COTS HPC systems”, ICML’13

# 昔と何が変わったのか？

- 0. データの増加、計算機的高速化
- 1. ネットワーク初期化手法
- 2. 最適化手法
- 3. 過学習回避の手法
- 4. その他ノウハウの蓄積

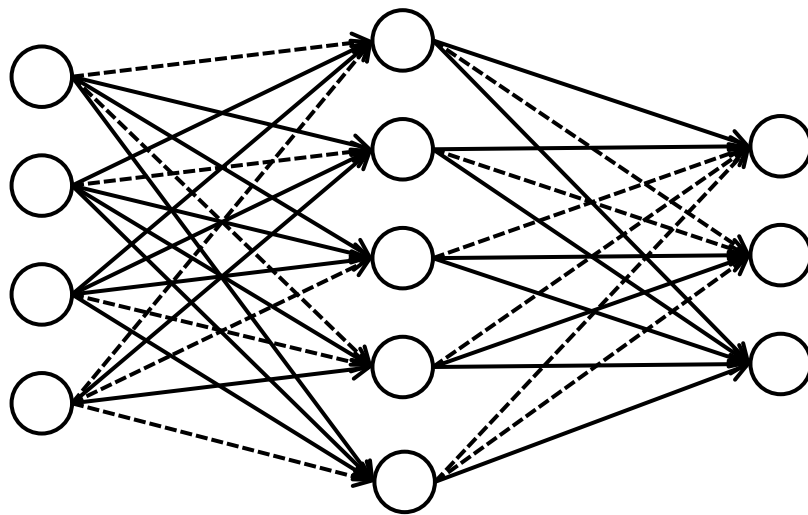
# Dropout

- 各訓練データのフィードバックの際に、確率0.5で中間ニューロンを無視（SGD前提）
- 認識時は全ニューロンを使うが、結合重みを半分ににする
- 多数のネットワークを混ぜた構造
  - 訓練データが異なるため、バギングと同様の効果（ただしパラメータは共有）
  - パラメータの共有による正則化の効果
- L2正則化と等価
  - [Wager et al., NIPS'13]
- 深層学習における最も大きなブレイクスルーの一つ



# 亜種

- Dropconnet [Wan et al., ICML'13]
  - ニューロンではなく、結合をランダムに落とす
  - Dropoutよりよいらしい？



- Standout [Ba et al., NIPS'13]
  - Dropoutで落とすニューロンをランダムでなく適応的に選択する

# Deep learning まとめ

- 深い構造でデータドリブンに全てを学習
  - なんでうまくいくのかまだよく分かっていないが、解析は着々と進行中  
(unsupervised pre-training, dropoutなど)
  - 同じ複雑さのモデルを表現する場合、浅いモデルよりもパラメータが少なく済む = 計算コストが軽くて済む！
- チューニングはとても大変
  - ハイパーパラメータの探索や、可視化の研究はホットなトピック
- まだ何も考えずに使えるレベルには至っていない
  - 各コンペでいい性能を出しているのは、ドメインに適した構造の識別的ネットワーク
  - もっとデータが増えれば様子が変わるかも？

# 勉強したい人向け

- ソフトウェア

- Theano

- <http://deeplearning.net/software/theano/>

- Pylearn2

- <http://deeplearning.net/software/pylearn2/>

- Cuda-convnet

- <https://code.google.com/p/cuda-convnet/>

- 論文・資料

- NIPS, ICML等のチュートリアル

- arXiv 上に最新の論文が多数投稿

# データサイエンス

~まとめ~

# この講義の目標（再）

- データを見て、考える習慣や、基本的な道具立て・センスを身につける
  - 実データをさわることを重視
  - 機械学習、数学もしっかり中身を理解
  - 各自の研究開発へ応用できるように
- 習うより慣れろ
  - プログラミングで言うところの Hello World
  - より発展的な勉強を自分で出来るようになる

# スキルセット

ハード	IT系スキル	RDBMS関連、SQL、Hadoop関連、JAVA、HDFS関連、MapReduce関連、Hive、pig
	分析系スキル	R、Python、Perl、Mahout、MADlib、Jubatus などの言語に関する知識と経験
		各種統計解析、各種機械学習に関する知識、SAS、SPSS、KXEN、KNIME などのツールに関する知識と経験
ソフト	ビジネス系スキル	業界・業務に関する知識、質問力、理解力、伝達力、説得力、プロジェクト推進能力

- どちらかというとなRは分析、Pythonは開発向きの印象
- この講義では7～8割方Pythonを使おうと思います。

# やってきたこと

- Python、R
- 統計基礎： 検定・推定
- 多変量解析・次元削減： PCA, FLDA, CCA等
- 回帰分析： 線形回帰、スパース線形回帰、一般化線形モデル
- 時系列分析： ARMA, ARIMA
- クラスタリング： k-means、EMアルゴリズム
- アソシエーション分析： アプリアリアルゴリズム
- クラス分類： 最小二乗識別、ロジスティック回帰、SVM
- アンサンブル学習： バギング、ブースティング (adaboost)

# データサイエンス：私が好きな定義

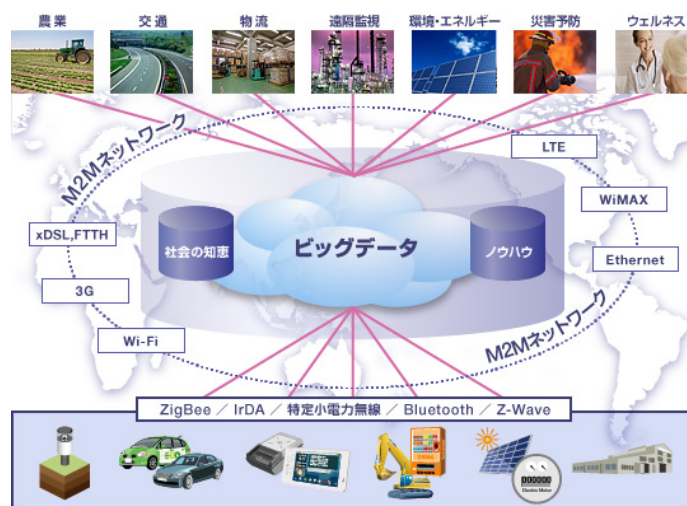
- データを収集し、分析に適した形に整え、**データにストーリーを語らせ、そのストーリーを他者に伝える**
  - Loukides, 「What is data science?」, 2010.
- ビジネス、システムの現状を理解した上で、データをもとに、**そもそも何をしたらよいのか**を提示する
- データ“**サイエンス**”と言われる所以

# データマイニングのプロセス

- CRISP-DM (CRoss-Industry Standard Process for Data Mining)
    - Business understanding
    - Data understanding
    - Data preparation
    - Modeling
    - Evaluation
    - Deployment
- 実際はここでつまづくことが多い...  
(もう少しやりたかった)
- アカデミアでは(主に)ここを極める

# 最後に

- いろいろなチャンスがある面白い時代
  - 実世界と情報世界が密に結合
    - M2M, IoT, ロボティクス, ...
  - 真にデータドリブンに世界を扱える時代が来る（かも）
    - 現在の研究分野はそのうち区別する意味がなくなる？



[http://jpn.nec.com/cloud/service/saas\\_common/m2m.html](http://jpn.nec.com/cloud/service/saas_common/m2m.html)

- 躍らされず、ただ斜に構えず、上手にビッグデータ時代を楽しんでください！