物理屋には分かる機械学習

富谷 昭夫 (IOPP in CCNU)

akio.tomiya_AT_mail.ccnu.edu.cn

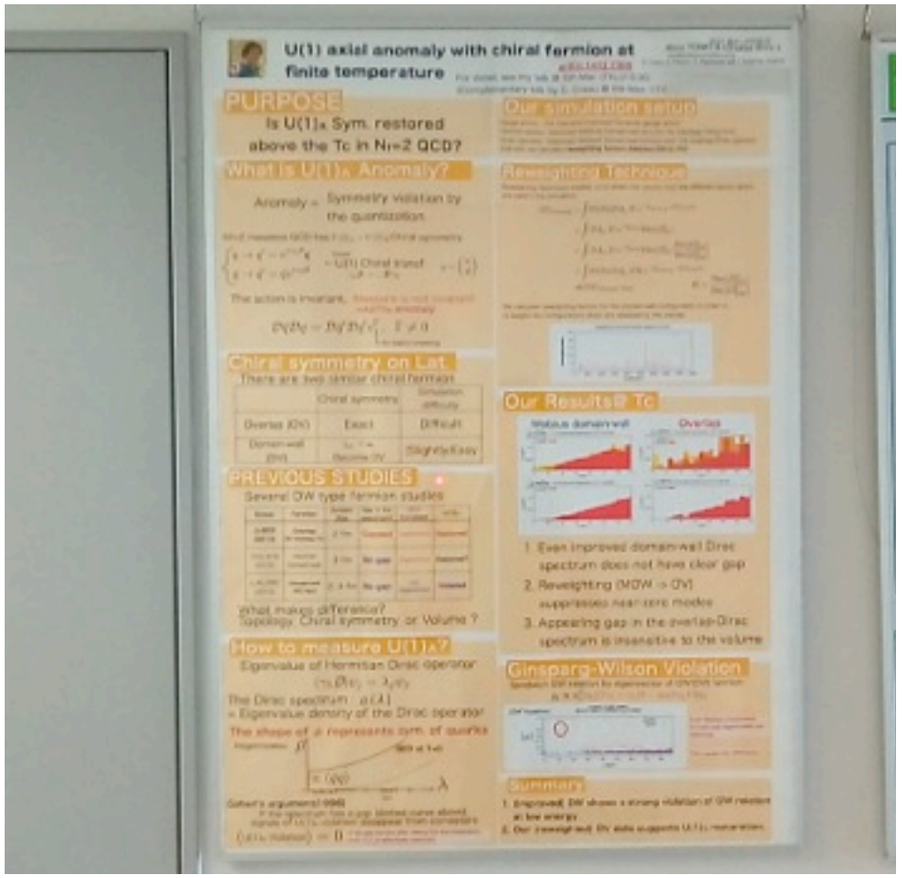
メッセージ

- 深層学習 (or ニューラルネット) は、魔法の道具ではなく、便利な数学的な道具。
- ニューラルネットは、データの背後にある相関を類推している。

だれ？

2015年に阪大素粒子論研究室で博士とった人です。

富谷 昭夫
中国、武漢の大学
(CCNU)でポスドク



このポスターの人です

やってること
興味のあること

QCDの相転移, ゲージ理論の相構造
エンタングルメントエントロピー, くりこみ群,
余剰次元模型など**を数値計算でやる。**
(物性出身なので物性も好きです)

Google

富谷 昭夫 | **検索**

なんで機械学習？

昔から興味があったので研究に使ってみました

中学の時から興味があったのですが、
ようやく理解できる数学知識とタイミングが揃ったので
理解して研究に使ってみました。

J. Phys. Soc. Jpn. 86, 063001 (2017) [4 Pages]

Next Article »

LETTERS

Detection of Phase Transition via Convolutional Neural Networks

Abstract | Full Text | References (35)

Full text: PDF (1295 kB)

Akinori Tanaka[1], Akio Tomiya[2],

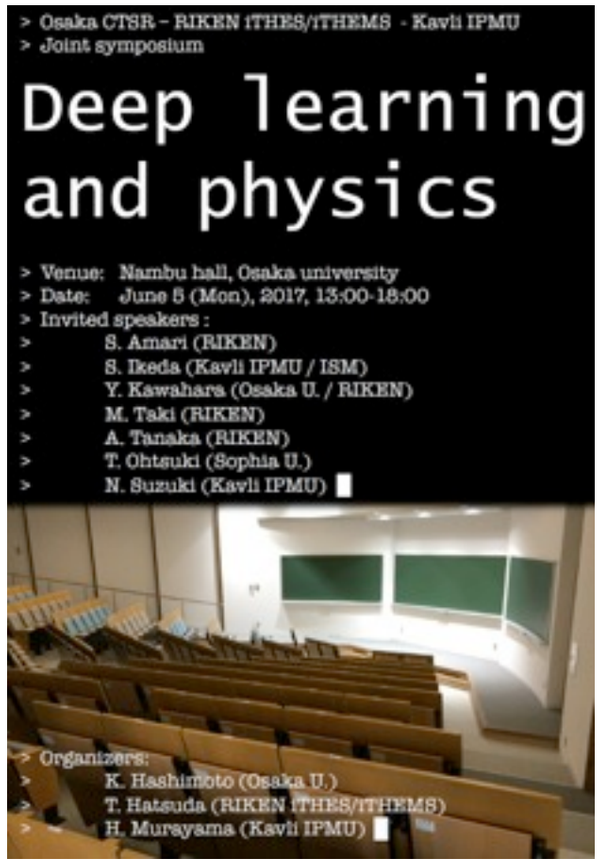
+ Affiliations

Received March 25, 2017; Accepted April 10, 2017; Published April 28, 2017

A convolutional neural network (CNN) is designed to study correlation between the temperature and the spin configuration of the two-dimensional Ising model. Our CNN is able to find the characteristic feature of the phase transition without prior knowledge. Also a novel order parameter on the basis of the CNN is introduced to identify the location of the critical temperature; the result is found to be consistent with the exact value.

DOI: http://dx.doi.org/10.7566/JPSJ.86.063001

研究の話は、
来週月曜の研究会(右図)で田中くんが話してくれます
(ので今日は、話しません)



A. Tomiya

今日話す事

- 目標: ニューラルネットとはどういうものかを(ざっくりと)理解する。
- 物理実験とフィッティング(最小二乗法)
- 線形代数と微積、そしてニューラルネット
- ベクトルとOne-hot表現
- たたみこみ、深層学習

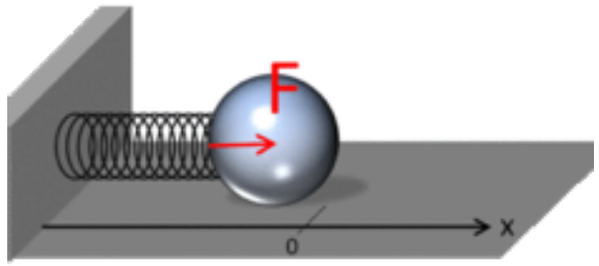
今日、話さない事

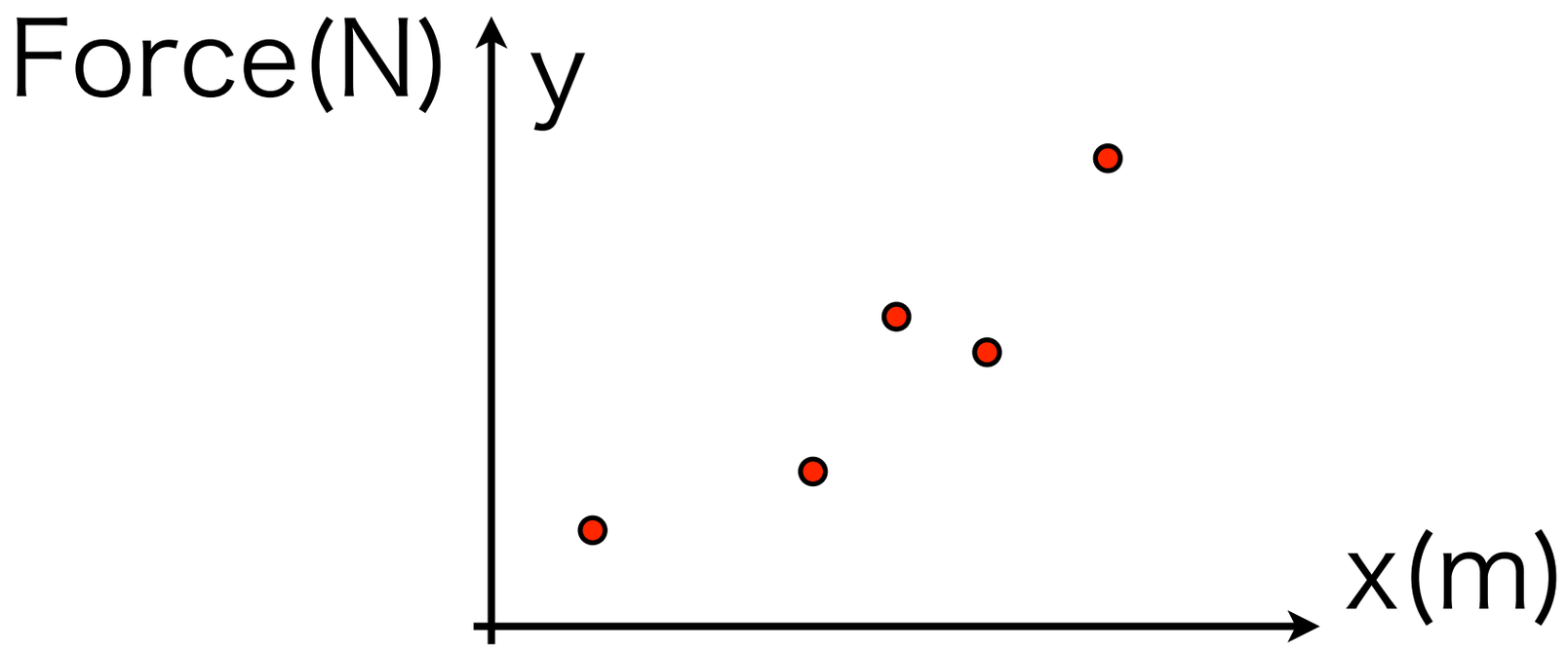
- (比較的)近年の発展：
  - バッチ正規化
  - 3DConvなど
  - 細かい数学的な話
  - RNN(Recurrent Neural Network)などのループ構造のあるもの。
  - Drop out, early stoppingなどのオーバーフィットを避ける技術
  - ボルツマン機械(画像、配位生成に使う)などの双方向のもの
  - Tensorflowなどの具体的なライブラリの使い方(Qiitaに良記事が沢山)。

ニューラルネットワークとは？

フィッティング→予言（内挿）

(例) フックの法則を確かめる実験をやったとしてみる。





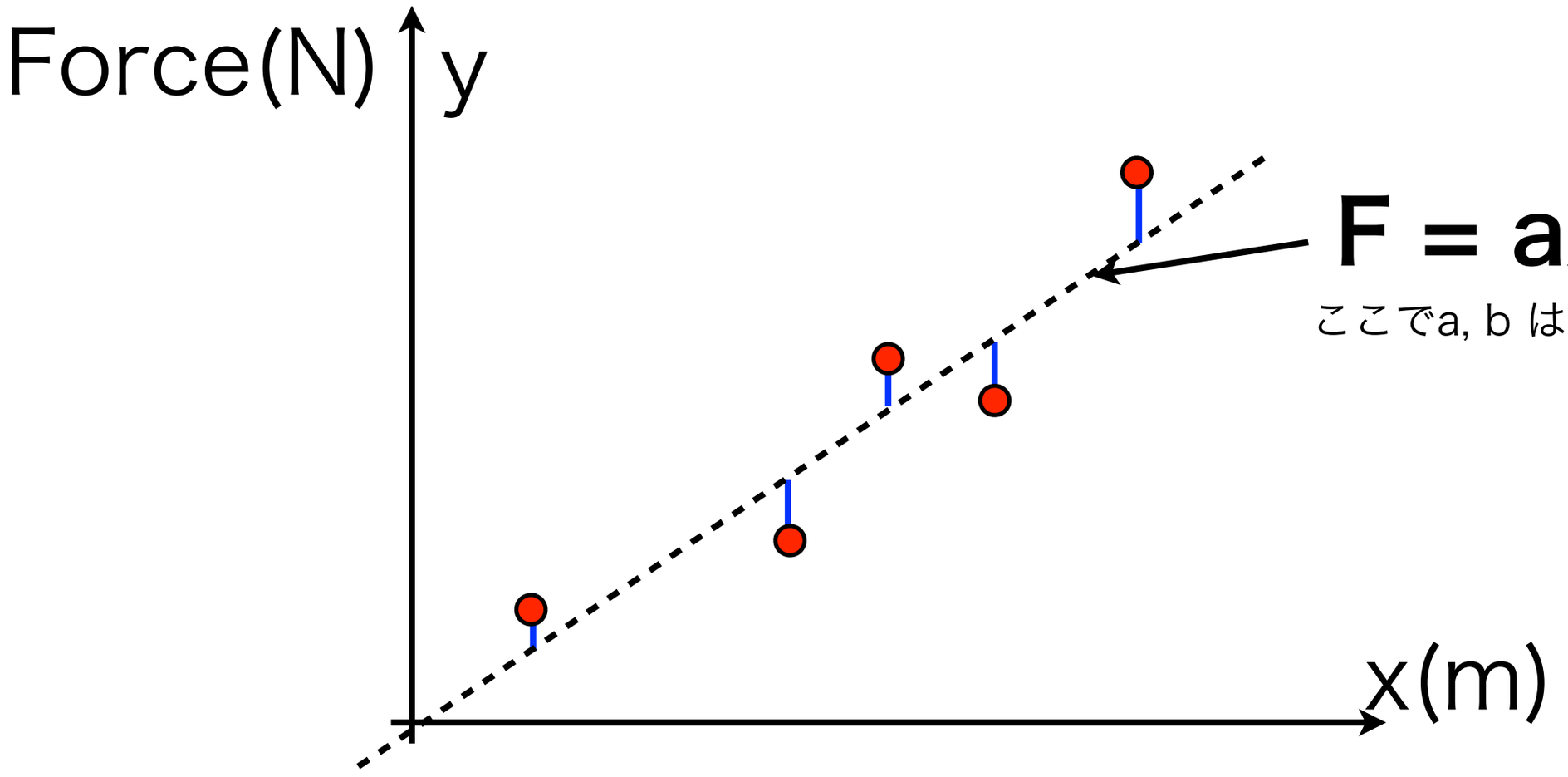
data set = {(x_0, y_0), $\underline{(x_1, y_1)}$, (x_2, y_2), ...}

この様な組をデータと呼ぶ

ニューラルネットワークとは？

フィッティング→予言（内挿）

最小二乗法を使うとバネ定数k がきまる。ここではもう少し
原始的に考えてみる



F = ax +b

ここでa, b は”フリーパラメータ”

E(1データ) ≡ ある点での直線と点の距離の二乗

“a” と “b”を調整し “E” を”最小化” => 直線が決定できる

この”E”は、誤差関数、損失関数(error/loss func)と呼ばれている

ニューラルネットワークとは？

フィッティング→予言（内挿）

E(1データ) ≡ ある点での直線と点の距離の二乗

$$\frac{\partial E(a,b)}{\partial a} = 0$$

$$\frac{\partial E(a,b)}{\partial b} = 0$$

最小二乗法では、↑の解を求める

ここでは、異なるアプローチを取る。以下をデータごとに以下を繰り返す(バッチ学習):

$$a_{\rm next} = a_{\rm prev} - \epsilon \left.\frac{\partial E}{\partial a}\right|_{\rm 1data}$$

$$b_{\rm next} = b_{\rm prev} - \epsilon \left.\frac{\partial E}{\partial b}\right|_{\rm 1data}$$

ε : 学習係数(小さい数)

沢山のデータに対して繰り返せば、もっともらしい値に近づいていく
（と、期待される）

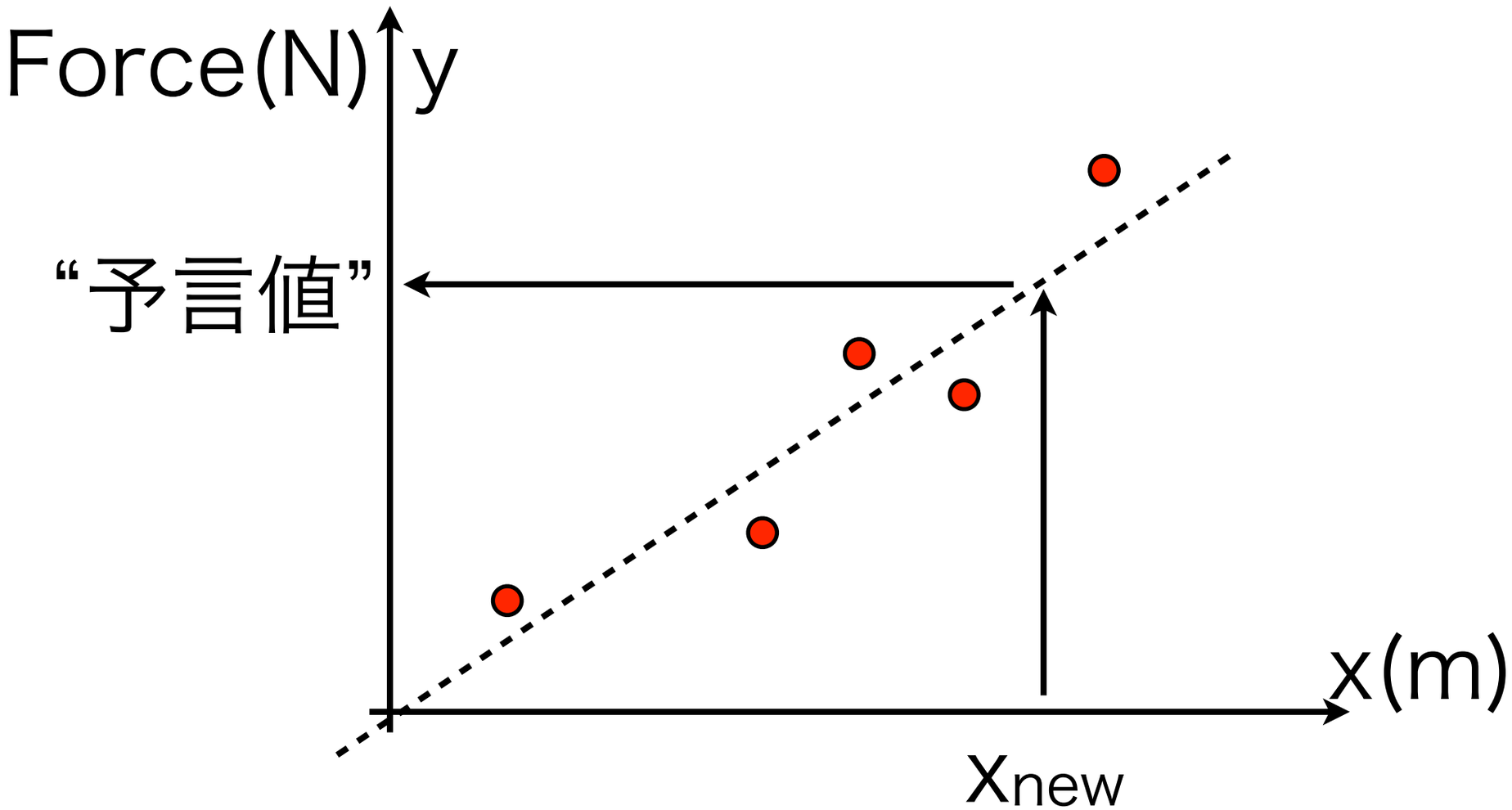
これは最急降下法 (正確には確率的勾配降下法)と呼ばれる
収束性の証明は甘利による. (c.f. 格子QCDで使う共役勾配法)

ニューラルネットワークとは？

フィッティング→予言（内挿）

もし直線を決定できれば,

新たな入力値“x_{new}” に対して予言を与えることができる



ニューラルネットワークがやっているのは、これの多次元版。

ニューラルネットワークは、2つの対象間の写像を見つける。

ニューラルネットワークとは？

フィッティング→予言（内挿）

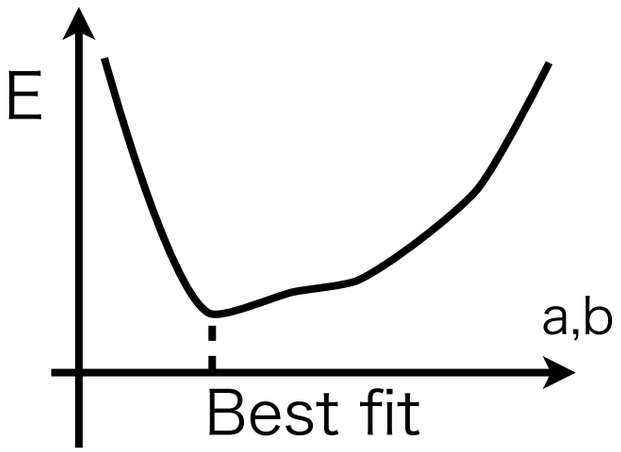
教訓(1/3):

$$F = ax + b \quad - (1)$$

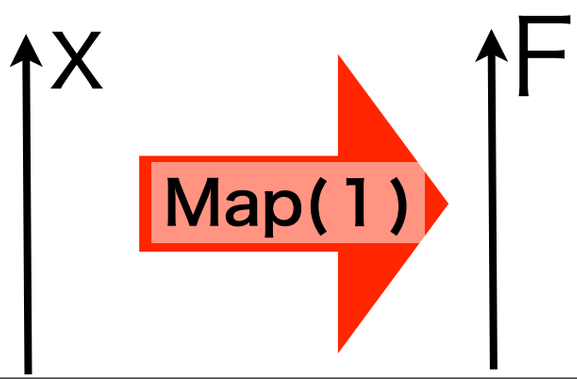
1. (1) 式は、２つのパラメータがある。パラメータを決めた後であれば、入力値を与えれば、予言値を出せる(外れることもある)。

E(1データ) ≡ ある点での直線と点の距離の二乗

2. エラー関数は非負。なので底をみつけるとそのパラメータが“ベストフィット”を与える。実は、距離の2乗ではなく、情報理論(情報幾何)で言うところのダイバージェンスであれば良い。(ベクトル解析の”発散”とは異なる)。ダイバージェンスの一例として相互エントロピー(カルバックライブラーダイバージェンス)がある。ダイバージェンスは、拡張したピタゴラスの定理を満たす(☆)



3. 式 (1) は、距離の空間(集合)から力の空間集合への写像を定めているとも見れる。この視点は後で使う。



(☆) 情報幾何学の基礎(藤原彰夫著)

ニューラルネットワークとは？

フィッティング→予言（内挿）

教訓(2/3):

4. パラメータは、以下の式で更新するとした
(Stochastic Gradient Descent 確率的勾配降下法)

$$a_{\mathrm{next}} = a_{\mathrm{prev}} - \epsilon \frac{\partial E}{\partial a}\Big|_{\text{1data}}$$

$$b_{\mathrm{next}} = b_{\mathrm{prev}} - \epsilon \frac{\partial E}{\partial b}\Big|_{\text{1data}}$$

ε : learning rate(small number)

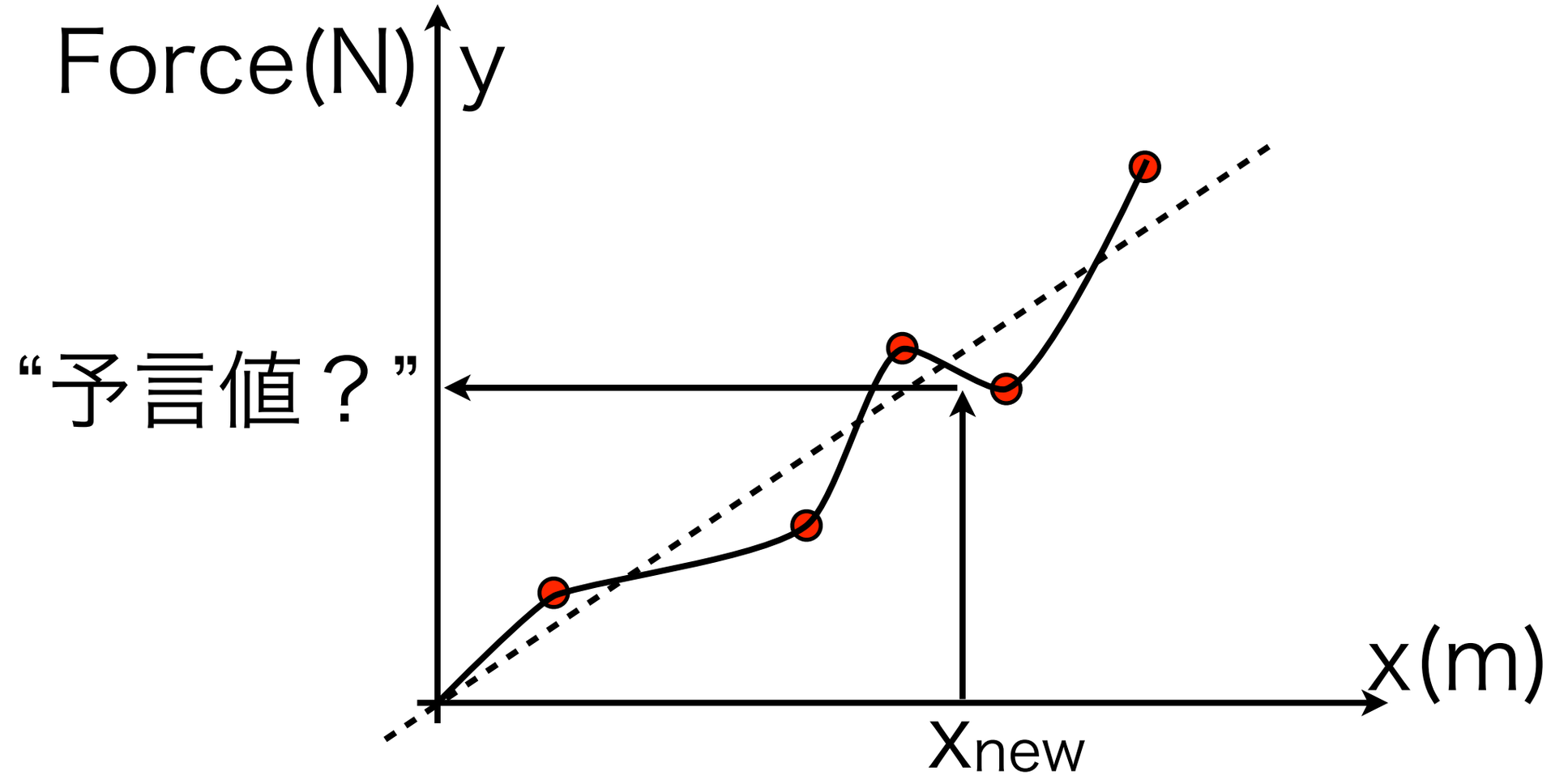
でも実は、この式をつかう必然性は無く、例えば1データだけでなく、
過去の履歴を使うなど 様々な改良案がある。
この手の更新式は、一般に”optimizer”と呼ばれ、現在も研究対象。
(効率的にパラメータ空間の底に辿りつくアルゴリズムの探求。)

ニューラルネットワークとは？

フィッティング→予言（内挿）

教訓(3/3):

5. フィット関数を変える(パラメータを適当に増やす)と、
全部の点を通る、いわゆるオーバーフィッティングをおこすことがある。
このときには、良い予言値を与えない。
ニューラルネットでも同様の問題が起こり得るが今回は説明しない。
(経験的な回避策がある)



線形写像と非線形写像

線形写像は、行列・ベクトル積とベクトル和

高次元のパラメータ空間に拡張するには、
下記のようにすれば良い

F = ax +b

→

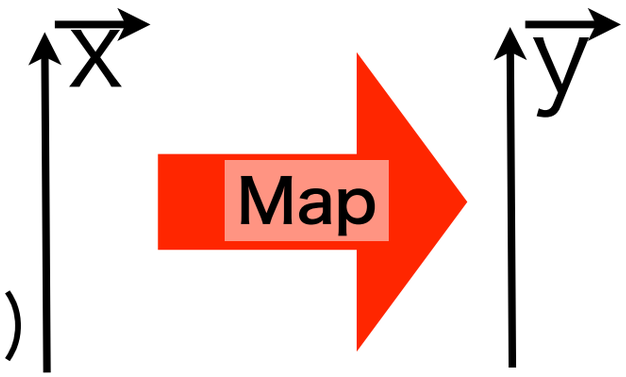
$$\vec{y} = W\vec{x} + \vec{b}$$

W : NxM 行列 **weight** と呼ばれる

x : M 成分ベクトル (入力)

b : N 成分ベクトル (バイアスと呼ばれる)

y : N 成分ベクトル (出力)



線形空間の演算は、行列、ベクトルの計算で表現できる。
(c.f. リー代数(環)の表現)

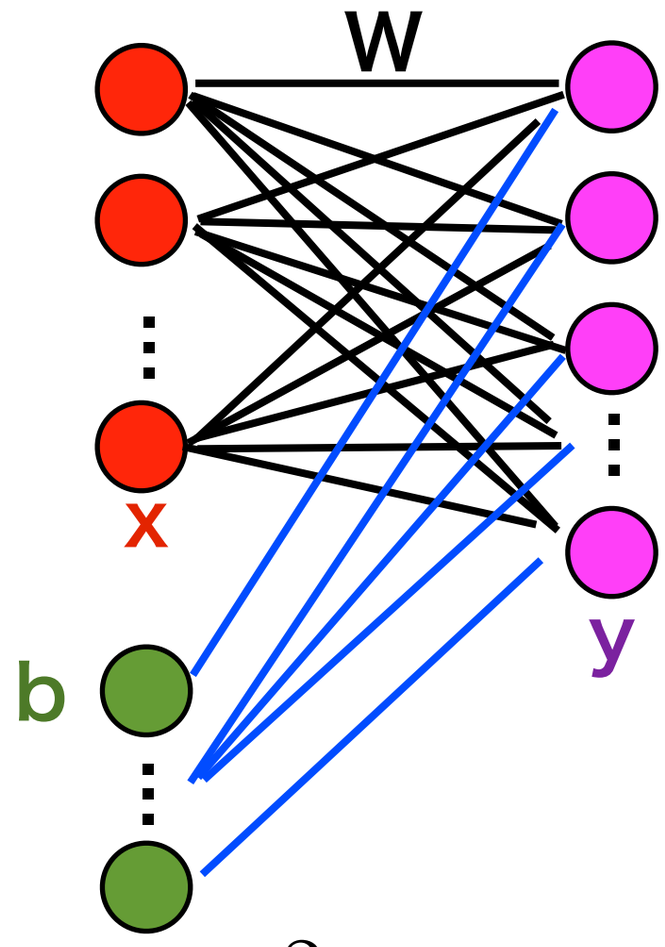
線形写像と非線形写像

線形写像は、行列・ベクトル積とベクトルの和で書ける

$$\vec{y} = W\vec{x} + \vec{b}$$

(誤差関数を減らすように
Wと b を調整する)

↑ 右辺から左辺へ読む。
↓ 左から右へ読む



隠れ層のない”ニューラルネット”
=> 単なる線形演算

$$E = |\vec{y} - \vec{y}_{\text{answer}}|^2$$

$$W_{ij}^{\text{next}} = W_{ij}^{\text{current}} - \epsilon \frac{\partial E}{\partial W_{ij}}$$

(ベクトルb についても同様)

線形写像と非線形写像

線形写像は、行列・ベクトル積とベクトルの和で書ける

線形演算からニューラルネットワークへ

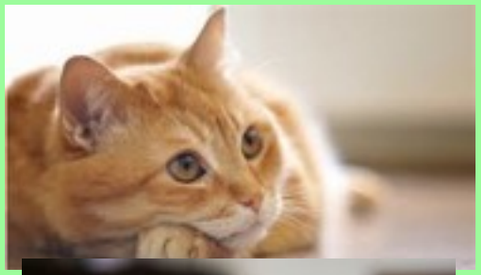
線形写像と非線形写像

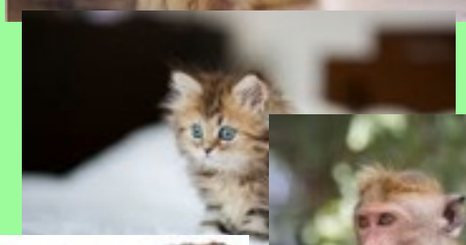
ニューラルネットワークは2集合間の写像を定める

ニューラルネットは2つの与えれた集合の間の写像を定める

(1)

動物の写真





(2)

名前

犬
猫
猿
...

(3)

Map

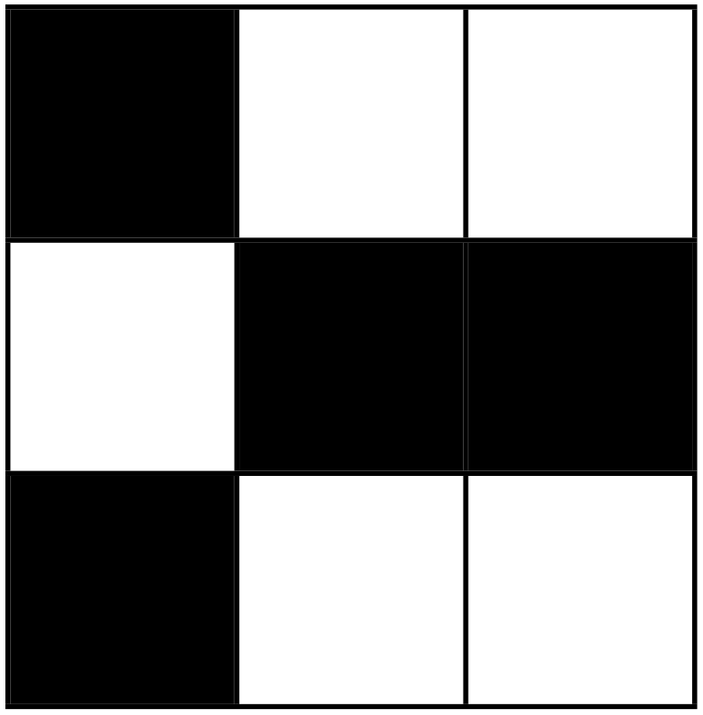
具体的にどうやって探すかが問題

線形写像と非線形写像

画像をベクトルとして見る

写像を定めるには、定量化(数字化)が必要。

(1)画像 → 1組の数字の列にする:



1	0	0
0	1	1
1	0	0

=> (100, 011, 100)

=> (1,0,0,0,1,1,1,0,0)

コレは、9次元空間のある点を指すベクトルになってる

線形写像と非線形写像

名前もベクトルとして見る

(2) 名前も同じく定量化が必要

「One-hot 表現」を用いる

まず、名前を単位ベクトルに関係づける

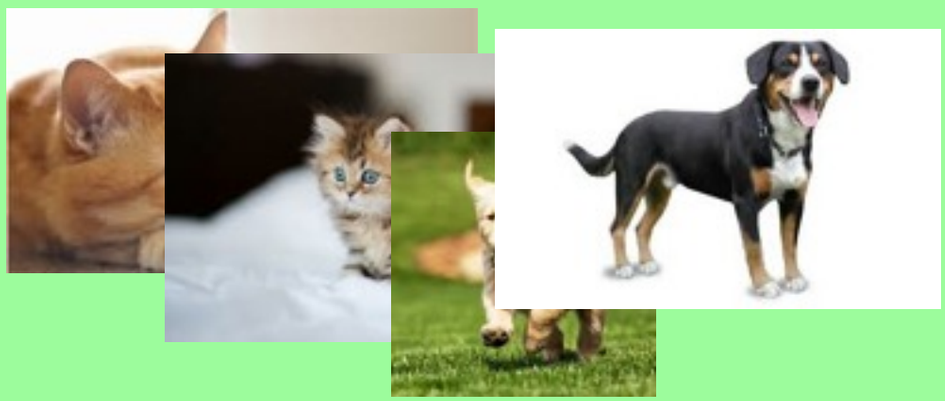
$$犬 = (1, 0, 0)$$
$$猫 = (0, 1, 0)$$
$$猿 = (0, 0, 1)$$

この場合、集合の元の数が3つであることを
知ってるため、3次元ベクトルの単位ベクトルで書く

線形写像と非線形写像

ニューラルネットワークは、2つの集合間の写像

(1) **動物の写真**



x=(1,0,0,0,1,1,1,0,0)

画像 -> ベクトル x

(3)

Map

(2) **名前**

Dog (1, 0, 0)

Cat (0, 1, 0)

...

名前 -> ベクトル y_{ans}

$$y = Wx + b$$

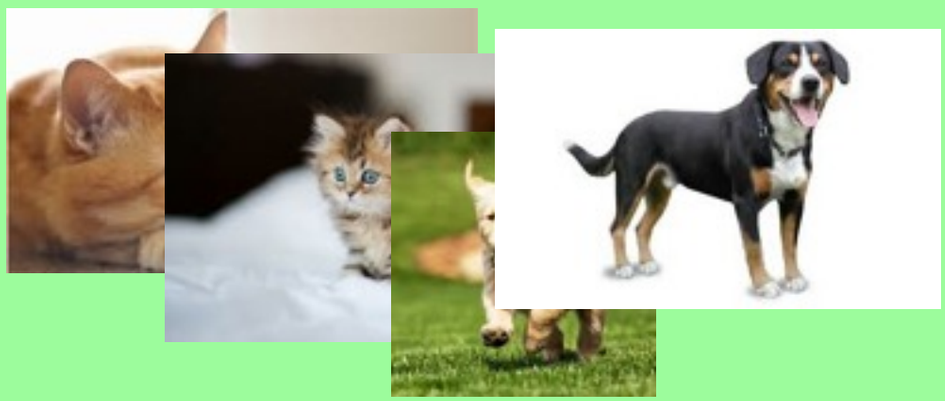
$$E = |y - y_{ans}|^2$$

コレは上手く行かない。

線形写像と非線形写像

ニューラルネットワークは、2つの集合間の写像

(1) **動物の写真**



x=(1,0,0,0,1,1,1,0,0)

画像 -> ベクトル x

(3) Map

(2) **名前**

Dog (1, 0, 0)

Cat (0, 1, 0)

...

名前 -> ベクトル y_{ans}

$$y = Wx + b$$

$$E = |y - y_{ans}|^2$$

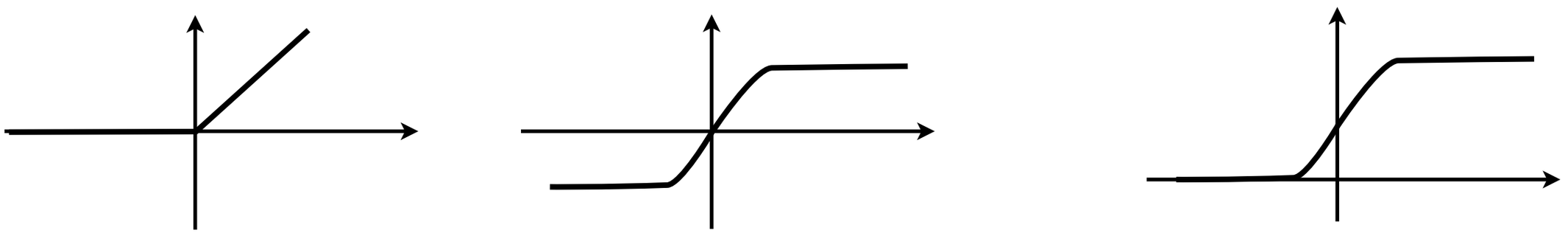
なぜならここのの写像は非線形だから。

-> 非線形性を導入しよう

線形写像と非線形写像

活性化関数は非線形性を与える

(2) **活性化関数** f(x)：非線形性を系にあたえる。



ReLu　　Tanh(x)　　sigmoid(x)

(適当な意味で)微分可能な非線形関数ならよい。

(微分可能性は、Eの微分に使う)

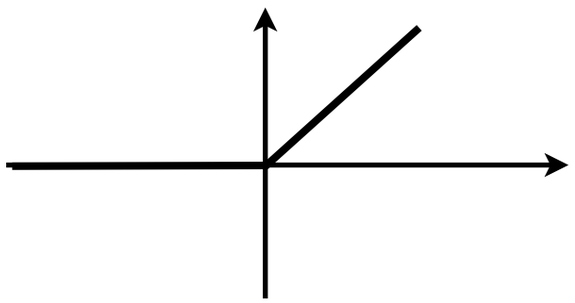
$$f(\vec{x}) \equiv \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{pmatrix}$$

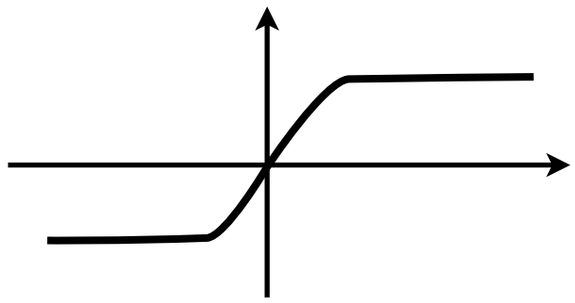
ここで、活性化関数は、element-wiseに作用する

線形写像と非線形写像

活性化関数は非線形性を与える

(2) **活性化関数** f(x)：非線形性を系にあたえる。

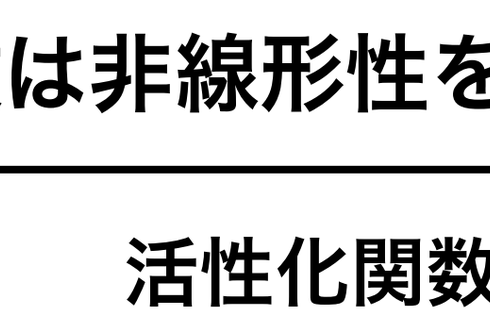




非線形関数(微分可能)

(1)

動物の写真



Map

(2) **名前**

犬　(1, 0, 0)

猫　(0, 1, 0)

...

するとこの写像は:

$$y = f(Wx + b)$$

当然、こうなるが、実際にはもう少し工夫する。

線形写像と非線形写像

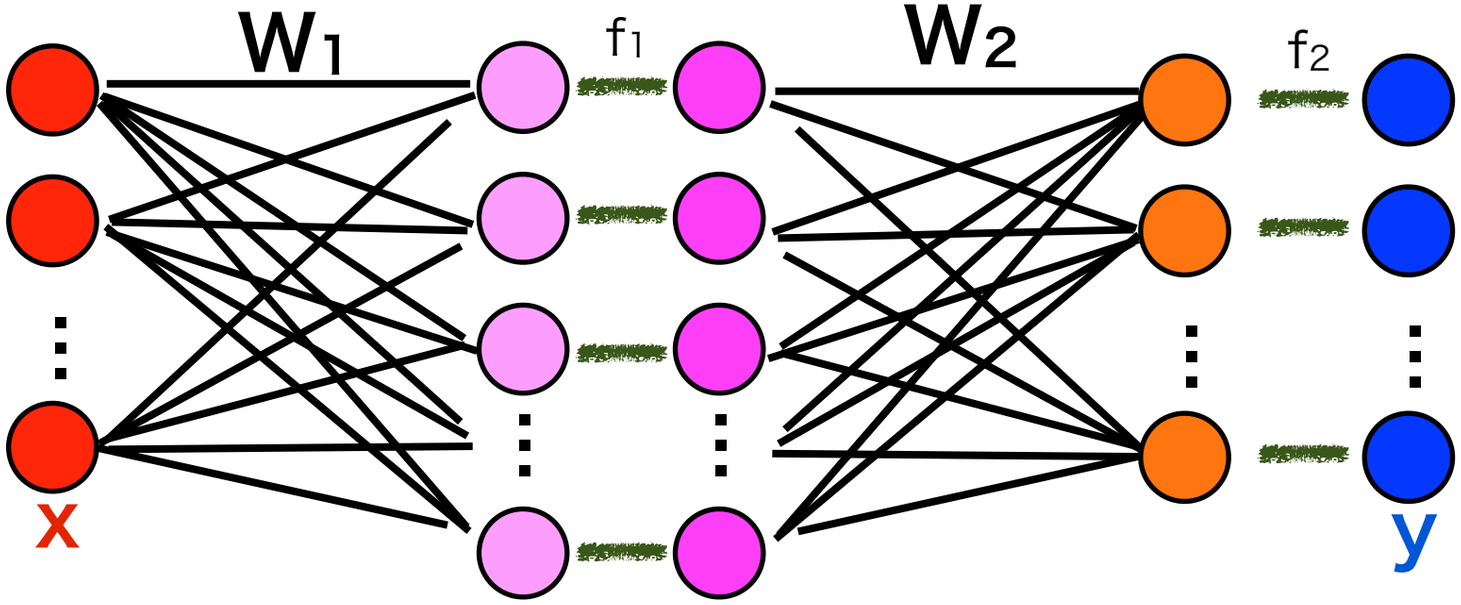
多層化して表現できる写像をリッチにする

例えば、多層化する。

$$y = f_2(W_2 f_1(W_1 \mathbf{x} + b_1) + b_2)$$

↑内側からよむ。

↓左から読む(慣例によりベクトルbは省略した)

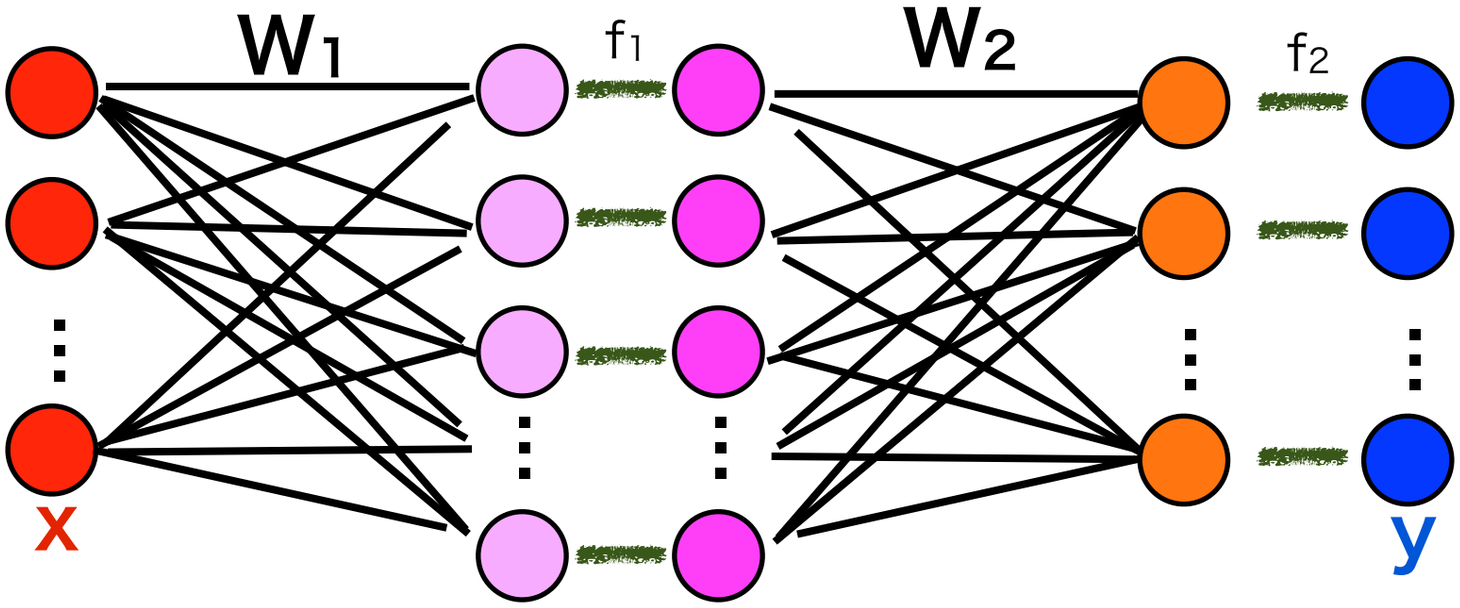


活性化関数は、各層で任意に選んで良い。

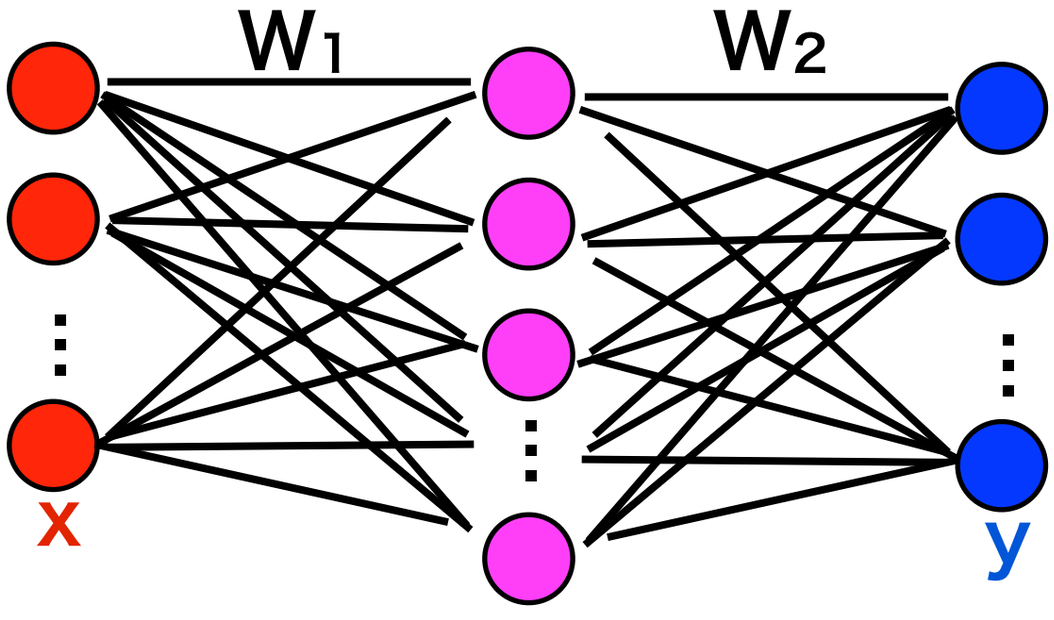
線形写像と非線形写像

多層化して表現できる写像をリッチにする

慣例として、活性化関数とバイアスは書かない。



≡



これを

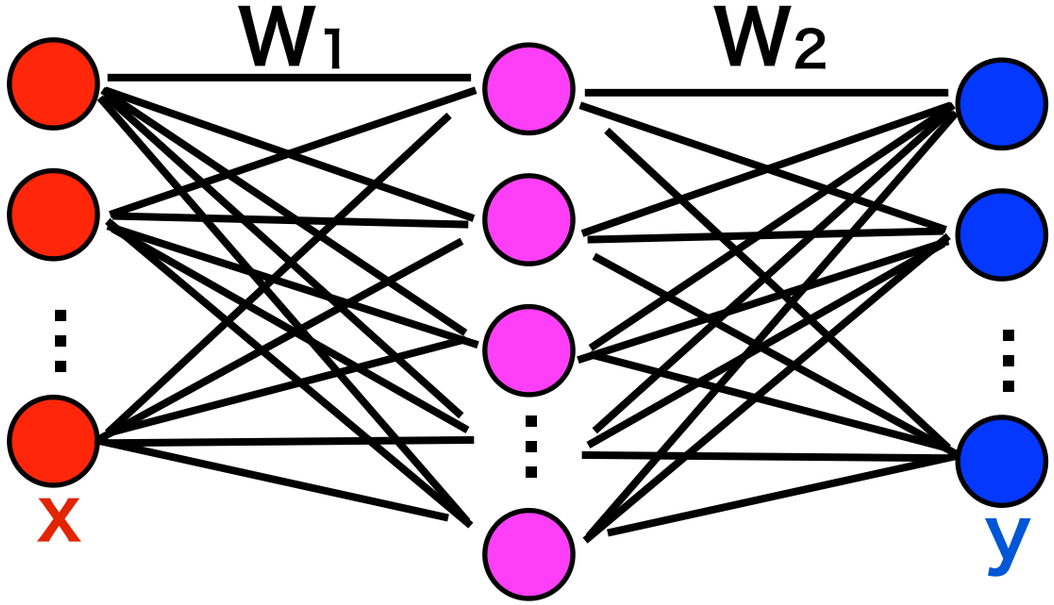
隠れ層1層の

ニューラルネット

と呼ぶ

線形写像と非線形写像

多層化して表現できる写像をリッチにする



隠れ層1層の
ニューラルネット

$\frac{\partial E}{\partial W_{ij}}$ $\frac{\partial E}{\partial b_i}$ をもってパラメータ更新するが、入れ子になってるので合成関数の微分が必要で
いわゆる逆誤差伝播とよばれる. これは|y-y_ans|という誤差が、左に向かって伝播するように
見える事からそう呼ばれる。(ここら辺はライブラリが自動的にやってくれる。)

実は、隠れ層と非線形関数の導入は本質的で、以下の定理を示すことができる。

Note: ニューラルネットワークの普遍性定理

(The universal approximation theorem: George Cybenko 1989)

もし、十分なユニット数があれば、隠れ層1層でもニューラルネットワークは、
任意の関数を近似可能である。
(次ページから説明)

普遍性定理

非線形関数を持つニューラルネットは任意の写像を近似できる

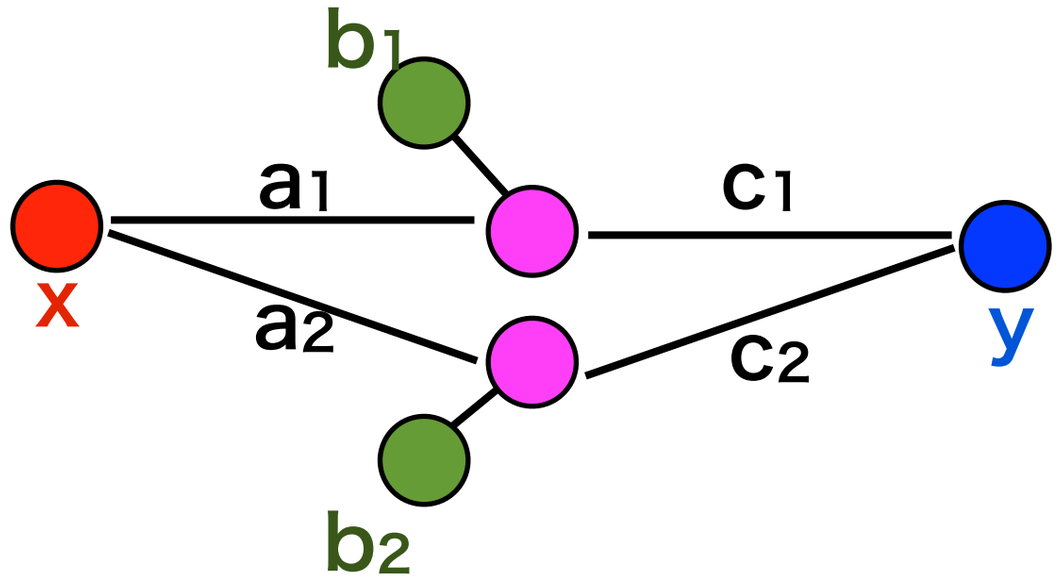
ニューラルネットワークの普遍性定理

(The universal approximation theorem: George Cybenko 1989)

(例)1パラメータ間の写像

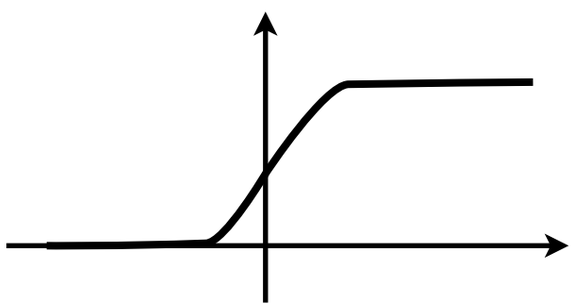
$$y=Cf(Ax+b)$$

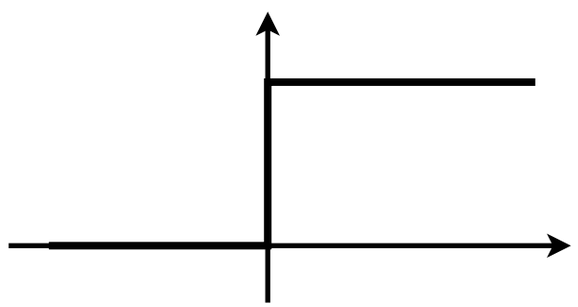
A: 2x1行列(要素a)、重み
C: 1x2行列(要素c)、重み



(微分可能ではないがシグモイド関数のトイモデルとして)階段関数を活性化関数として選ぶ。ここは本質ではない。

シグモイド





階段関数

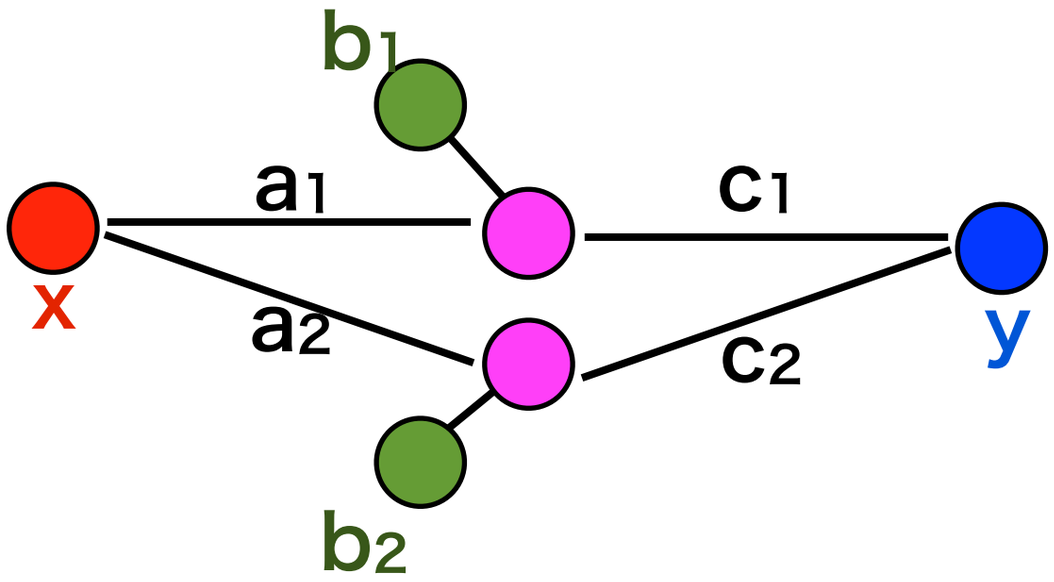
普遍性定理

非線形関数を持つニューラルネットは任意の写像を近似できる

(例)1パラメータ間の写像

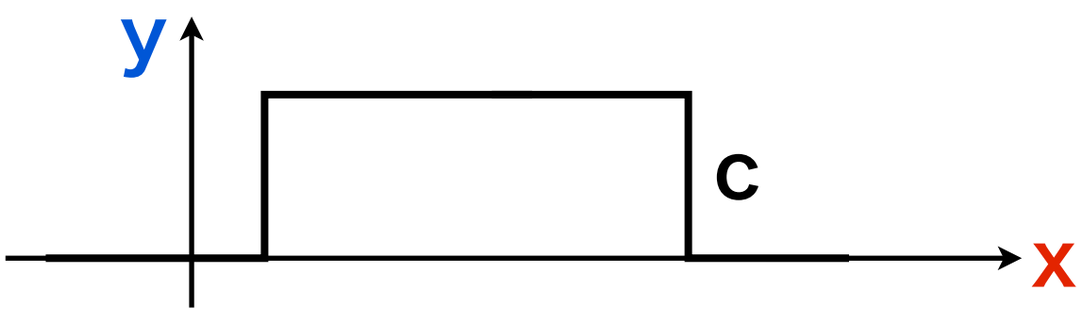
$$y=Cf(Ax+b)$$

重み　A: 2x1行列(要素a)
　　C: 1x2行列(要素c)



トイモデルとして、
活性化関数fを階段関数
と取った

事実1:　活性化関数が階段関数のとき、隠れ層の要素が2個、A=1,-1、c 共通と取り、bを選ぶと、以下の形を任意の場所、任意の幅(bで指定)、任意の高さ(cで指定、負でも可)で作れる。



$$y=cf(x+b_1)+cf(-x+b_2)$$

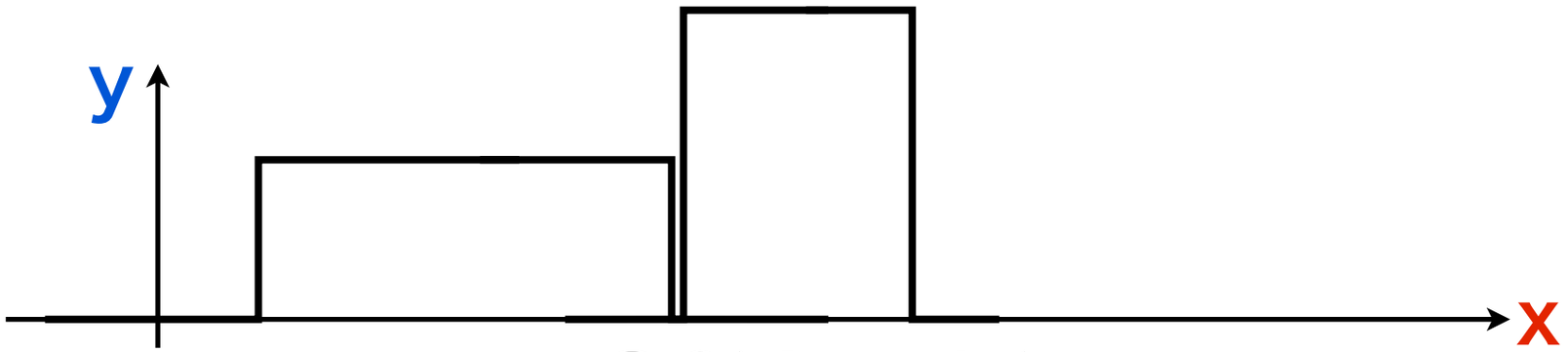
fは階段関数

普遍性定理

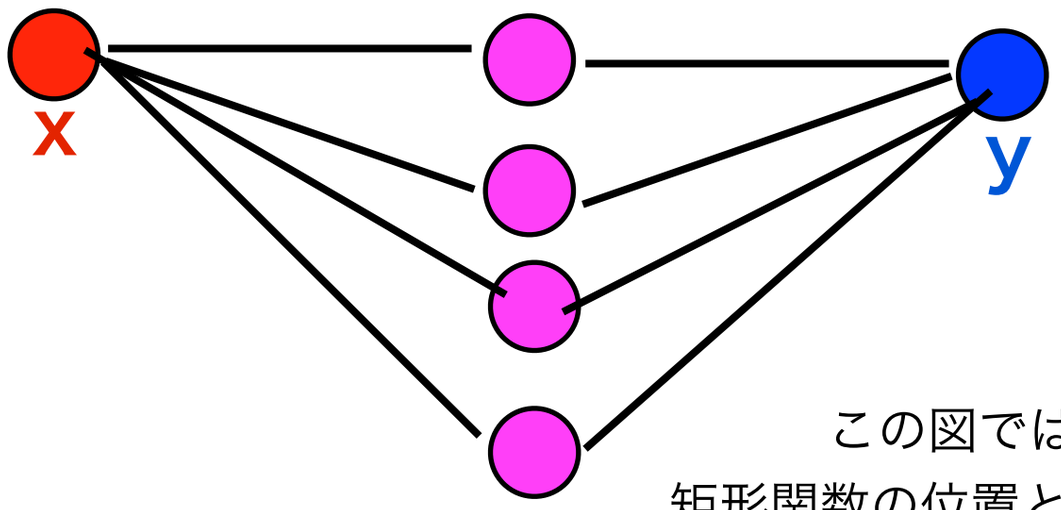
非線形関数を持つニューラルネットは任意の写像を近似できる

(例)1パラメータ間の写像

事実2:　隠れ層の数を増やすと、bの要素も増える。任意の位置、幅、高さの大量の矩形型の関数が作れ、それらは、<u>重ね合わされる</u>



y=Cf(Ax+b)



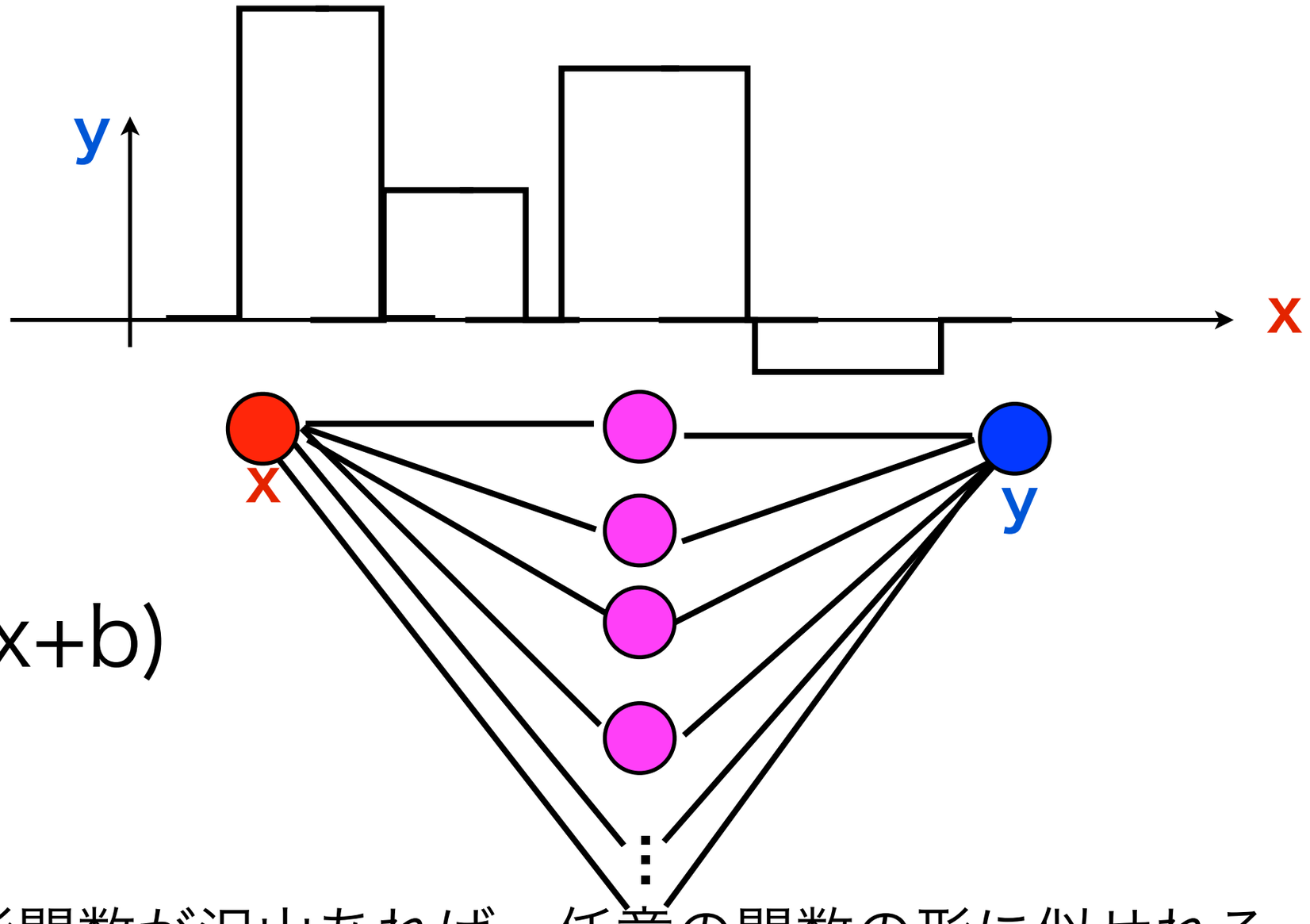
この図では、bは省略したが、
矩形関数の位置と幅を指定しているので必須

普遍性定理

非線形関数を持つニューラルネットは任意の写像を近似できる

(例)1パラメータ間の写像

事実1+事実2: 隠れ層の要素が沢山 = 矩形関数沢山



y=Cf(Ax+b)

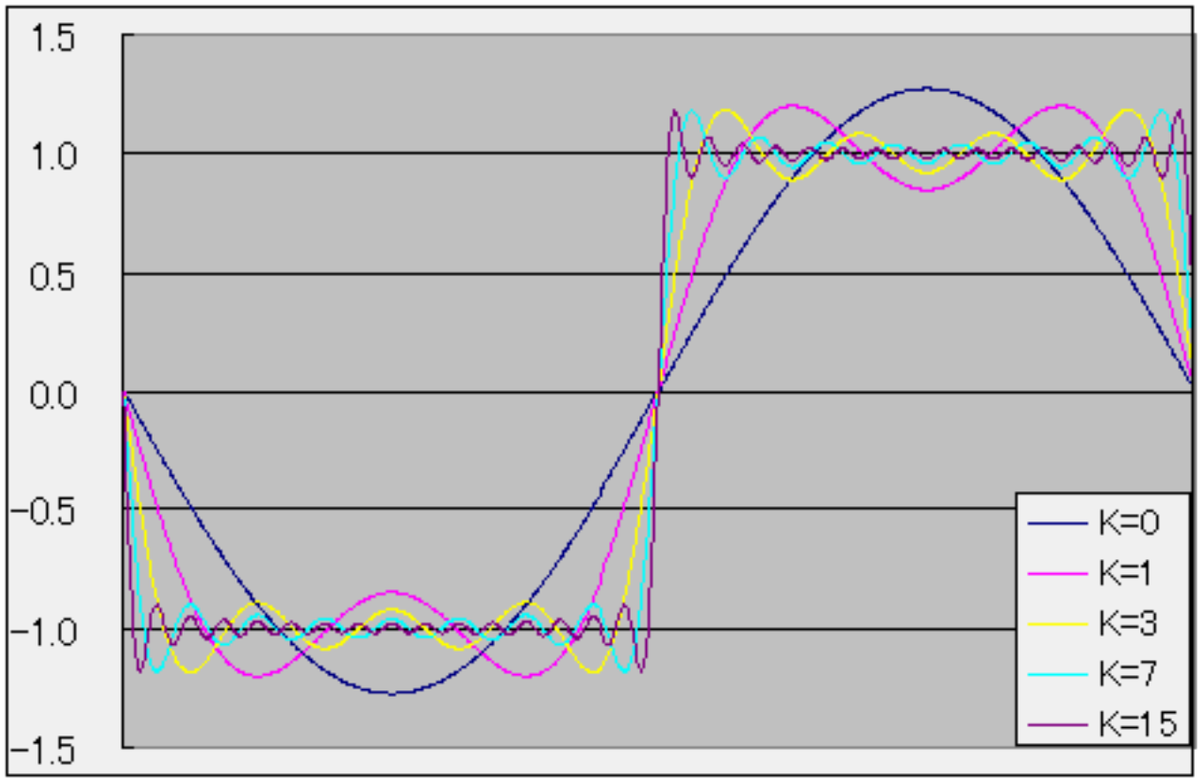
矩形関数が沢山あれば、任意の関数の形に似せれる
= ニューラルネットワークの普遍性定理

(普遍性定理は、高次元の関数(写像)でも、成立する。)

普遍性定理

非線形関数を持つニューラルネットは任意の写像を近似できる

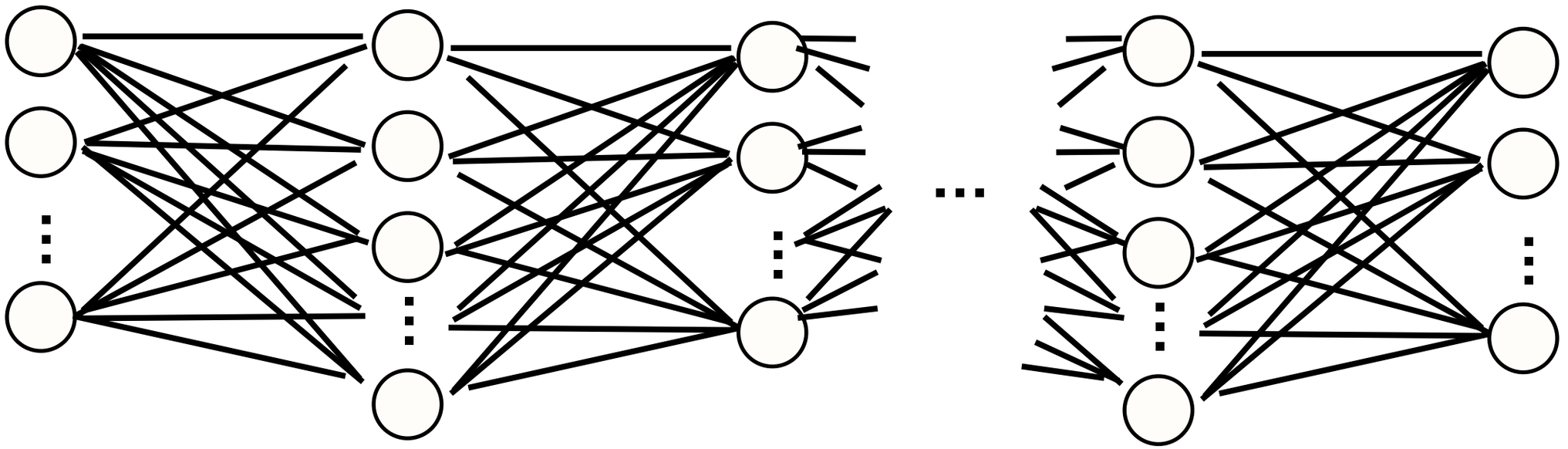
直感的には、フーリエ展開に似ている。
沢山のsin(x)とcos(x)があれば、周期関数は上手く近似(展開)できる(重ね合わせ)。
(いまは、矩形関数の重ね合わせで関数の形を表現していた。)



フーリエ展開も数学的な証明は、色々むずかしい。
(フーリエ変換となると、測度0の点とかを真面目に考える必要がある)
だが物理屋は、波の重ね合わせ(もしくは、スペクトル分解)として理解できてる。

Deep learning(深層学習)

たくさん積む。



沢山積んだ物は、ディープニューラルネットと呼ばれる。より強力。これが深層学習。

Note: 実のところ、深層にするというのは、そんなに賢い方法とされてこなかった。これは逆誤差伝播での合成関数微分が積で書かれるため深層化すると
異常に小さくなったり大きくなったりするため (勾配消失/発散問題)
こうなってしまうと、学習(パラメータフィット)が上手くいかなくなる。

この問題は、活性化関数の選び方やバッチ正規化などの手法で解決済であるが今回はとばします。

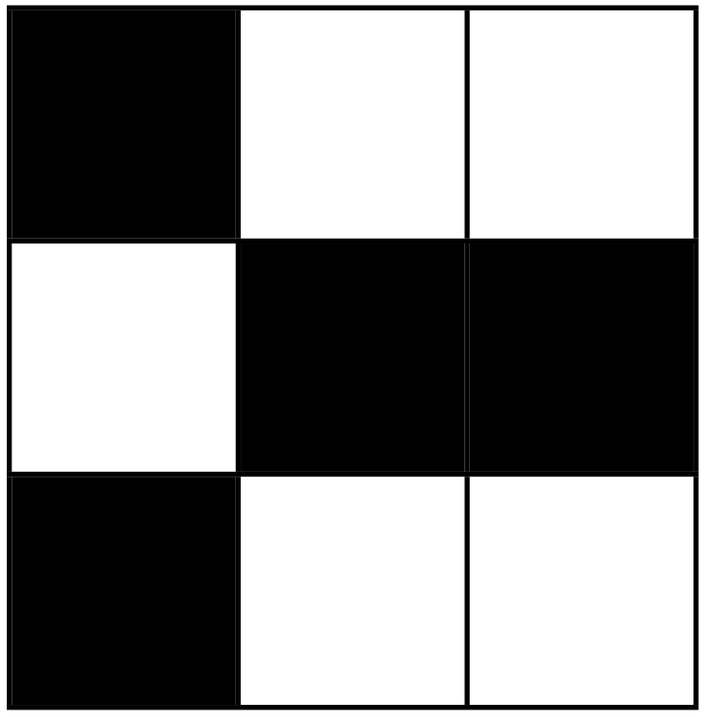
追記(6/3)

- 普遍性定理＝隠れ層が1層でも任意の写像を近似できる
- 深層化＝深い方が良い
- 大野木さんから「↑にテンションがある気がする。」という質問をいただきました。分からないので少し調べてみました。
- https://www.quora.com/What-is-the-implication-of-the-Universal-Approximation-Theorem-over-deep-learning-methodology
- ここでの回答が正しいとすると、層を深くする事と層の数を増やすことには関係がない、という事だそうです。
- また、 https://arxiv.org/abs/1608.08225 の様な論文も見つけました。物理の言葉でなぜdeepが良いか、を論じているようです。

たたみこみ

たたみこみ = 局所的な情報を拾う

画像から、ベクトルを作る時、



1	0	0
0	1	1
1	0	0

=> (1,0,0,0,1,1,1,0,0)

近い点同士が離れてしまって、

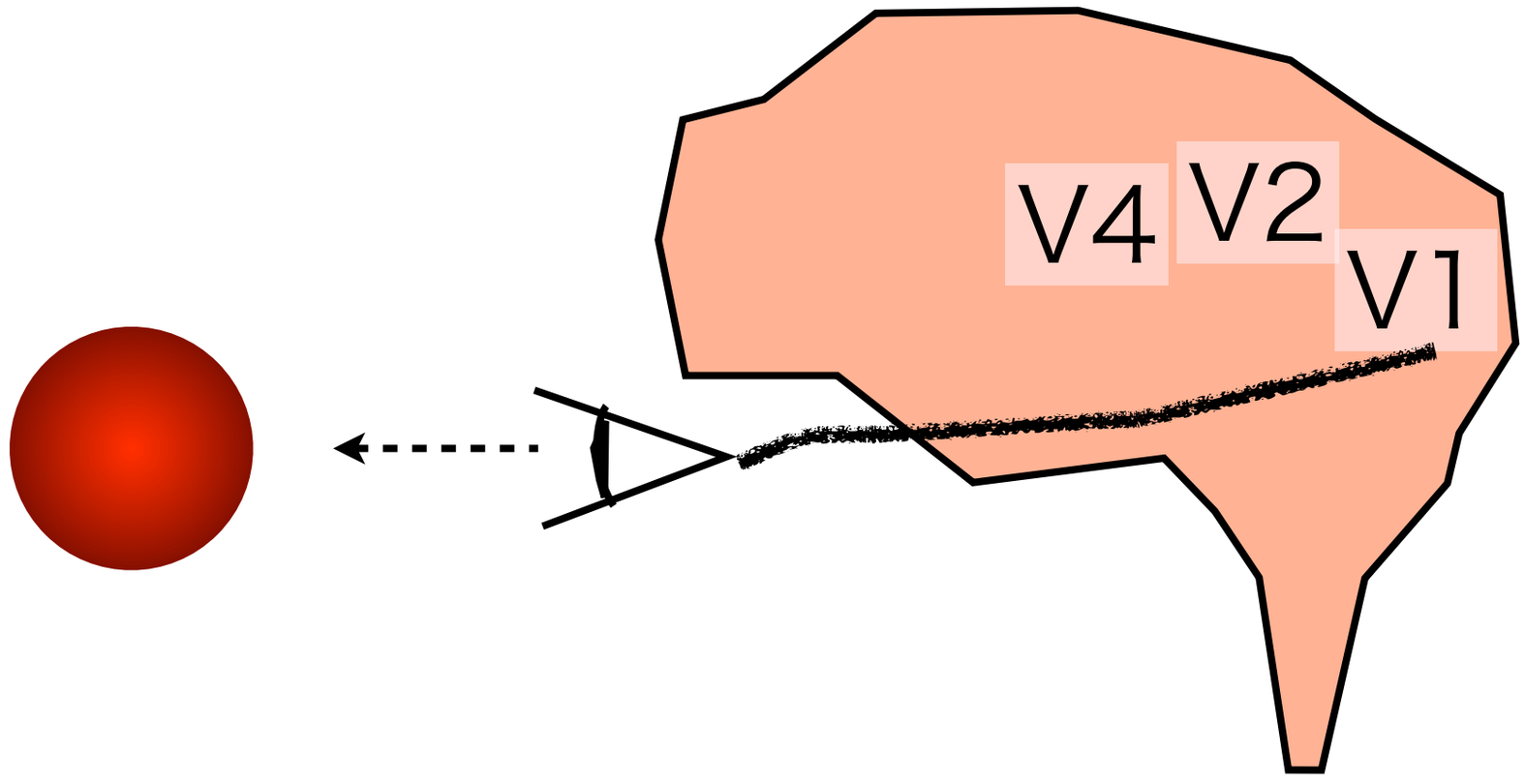
上手くニューラルネットワークが学べない事があった。

→「たたみこみ」で解決

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

歴史的には、動物の視覚野をモデルにしたネットワークモデル



比較して、たたみこみじゃない層（これまでに説明したニューラルネットワーク）は
Fully connected、全結合 とかdens と呼ばれる

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	0	1	1	0
0	1	0	1	1	0
0	0	0	1	1	1

画像

x

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

フィルター
(Fも学習対象)
学習過程では
サイズは固定。

$= 1F_1+0F_2+0F_3$
$+ 0F_4+1F_5+0F_6$
$+ 0F_7+0F_8+1F_9$

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

たたみこまれた
結果

Element-wiseな掛け算と和

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	0	1	1	0
0	1	0	1	1	0
0	0	0	1	1	1

画像

x

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

フィルター
(Fも学習対象)
学習過程では
サイズは固定。

$= 0F_1+0F_2+0F_3$
$+ 1F_4+0F_5+0F_6$
$+ 0F_7+1F_8+1F_9$

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

たたみこまれた
結果

Element-wiseな掛け算と和

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	0	1	1	0
0	1	0	1	1	0
0	0	0	1	1	1

画像

x

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

フィルター
(Fも学習対象)
学習過程では
サイズは固定。

$= 0F_1+0F_2+0F_3$
$+ 0F_4+0F_5+0F_6$
$+ 1F_7+1F_8+1F_9$

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

たたみこまれた
結果

Element-wiseな掛け算と和

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	0	1	1	0
0	1	0	1	1	0
0	0	0	1	1	1

画像

x

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

フィルター
(Fも学習対象)
学習過程では
サイズは固定。

$= 0F_1+0F_2+0F_3$
$+ 0F_4+0F_5+0F_6$
$+ 1F_7+1F_8+0F_9$

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

たたみこまれた
結果

Element-wiseな掛け算と和

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	0	1	1	0
0	1	0	1	1	0
0	0	0	1	1	1

画像

x

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

フィルター
(Fも学習対象)
学習過程では
サイズは固定。

$= 0F_1+1F_2+0F_3$
$+ 0F_4+0F_5+1F_6$
$+ 0F_7+0F_8+0F_9$

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

たたみこまれた
結果

Element-wiseな掛け算と和

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	0	1	1	0
0	1	0	1	1	0
0	0	0	1	1	1

画像

x

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

フィルター
(Fも学習対象)
学習過程では
サイズは固定。

$= 1F_1+0F_2+0F_3$
$+ 0F_4+1F_5+1F_6$
$+ 0F_7+0F_8+1F_9$

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

たたみこまれた
結果

Element-wiseな掛け算と和

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	0	1	1	0
0	1	0	1	1	0
0	0	0	1	1	1

画像

x

F_1	F_2	F_3
F_4	F_5	F_6
F_7	F_8	F_9

フィルター
(Fも学習対象)
学習過程では
サイズは固定。

$= 1F_1+0F_2+0F_3$
$+ 0F_4+1F_5+1F_6$
$+ 0F_7+0F_8+1F_9$

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

たたみこまれた
結果

ローカルな情報を拾っている

Element-wiseな掛け算と和

たたみこみ

画像解析で使うフィルター(物理屋的には離散ラプラス演算子等)に対応

たたみこまれた
結果

r_1	r_2	r_3	r_4
r_5	r_6	r_7	r_8
r_9	r_{10}	r_{11}	r_{12}
r_{13}	r_{14}	r_{15}	r_{16}

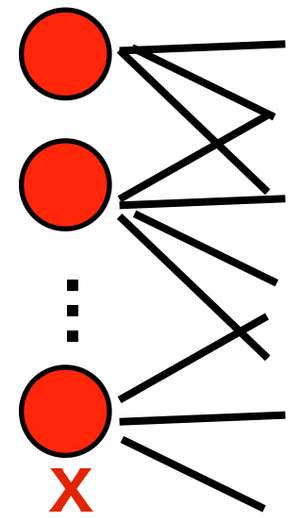
Flatten

大雑把に言って

2つの選択肢

r_1 ⋮ r_{16}

Input



Fully connected
全結合

more conv.

F'_1	F'_2	F'_3
F'_4	F'_5	F'_6
F'_7	F'_8	F'_9

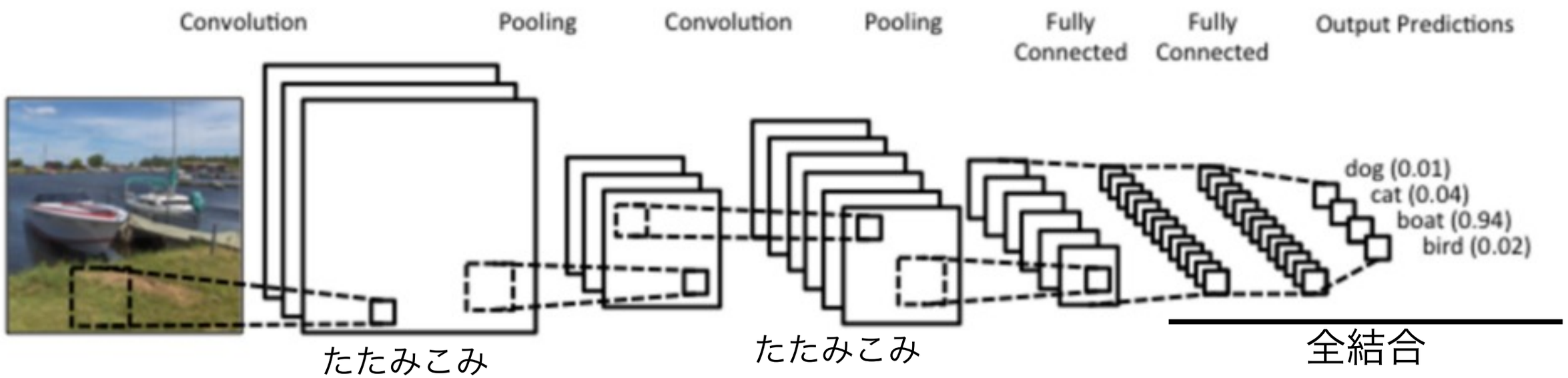
→ ...

※他にもPooling 等がある

深層学習

たたみこみ+ たたみこみ+ たたみこみ+…

深層学習＝「深層化＋たたみこみ+全結合」



=普通のニューラルネット

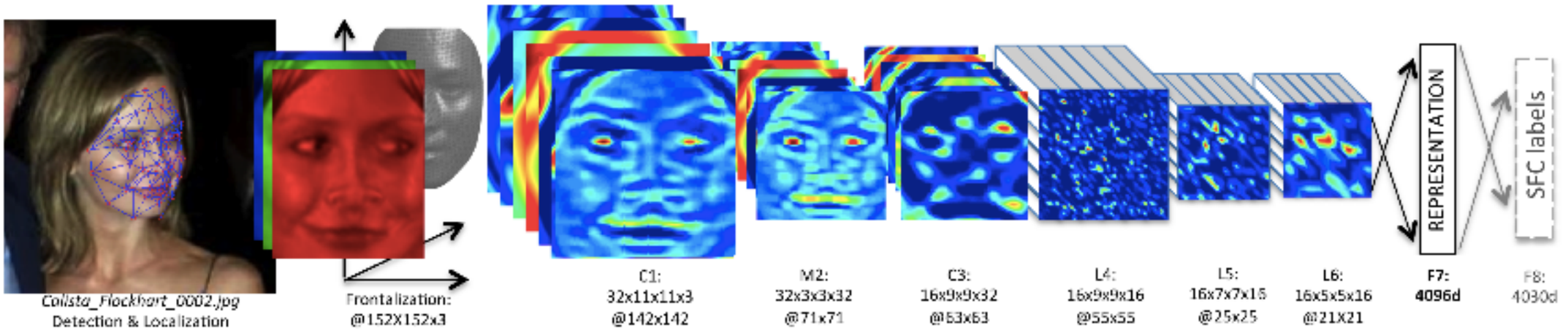
 https://cdn-images-1.medium.com/max/1000/1*N4h1SgwbWNmtrRhszM9EJg.png から引用 A. Tomiya

深層学習

たたみこみ+ たたみこみ+ たたみこみ+…

たたみ込みは、何層もつなげても良い。

深ければ、深いほど何か抽象的なモノを学んでる様子…？



[Y.Taigman 2014]

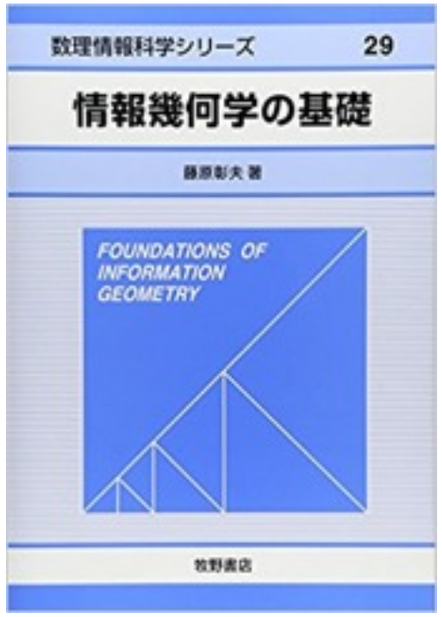
このあたりは、深層学習がうまくいく理由と合わせて
あまり理解が進んでいない(と思います)

今日のまとめ

- 目標: ニューラルネットとはどういうものかを(ざっくりと)理解する。
- 物理実験とフィッティング(最小二乗法)
- 線形代数と微積、そしてニューラルネット
- ベクトルとOne-hot表現
- たたみこみ、深層学習

参考文献

情報
情報幾何関連



機械学習関連

等

無料オンラインテキスト「ニューラルネットワークと深層学習」(普遍性定理の所)
https://nnadl-ja.github.io/nnadl_site_ja/
Slide share上の『数学カフェ 「確率・統計・機械学習」』のスライド
あと、Qiita の記事多数！