iTHEMS RIKEN interdisciplinary Theoretical & Mathematical Sciences

RIKEN

深層学習の汎化に関する数理的研究の進展

瀧 雅人 (Masato Taki)

RIKEN, iTHEMS

2019.3/5

統計物理学懇談会

@学習院大学

iTHEMS RIKEN interdisciplinary Theoretical & Mathematical Sciences

RIKEN

深層学習の汎化に関する数理的研究の進展

物理

瀧 雅人 (Masato Taki)

RIKEN, iTHEMS

2019.3/5

統計物理学懇談会

@学習院大学

数値

理論研究　アルゴリズム研究　新手法

応用

数学

理論研究　アルゴリズム研究　新手法

応用

知りたいこと

- 性能の秘密は？ 表現能力の秘密は？
- 超過剰なパラメータを持つのに逆にうまくいくのはなぜ？
- 結局何が他のモデルと大きく違うものにしているのか？

などなどたくさん

統計的学習

自然科学的・機械学習的世界観

$$\vec{x} \sim P(\mathrm{x})$$

統計的学習

自然科学的・機械学習的世界観

$$\vec{x} \sim P(\mathrm{x})$$

観測値

自然界のstochastic
な過程

観測データから、分布に関する情報を知りたい：

統計的学習

自然科学的・機械学習的世界観

$$\vec{x} \sim P(\mathrm{x})$$

観測値

自然界のstochastic
な過程

観測データから、分布に関する情報を知りたい：

- 分布そのもの
- 統計量、各種期待値など
- ランダム性を取り除いた「決定論的」なパターン、法則
- ・・・

Supervised Learning

$$\vec{x},\, \vec{y} \sim P(\mathrm{x}, \mathrm{y})$$

Supervised Learning

$$\vec{x},\ \vec{y}\ \sim\ \boldsymbol{P}(\mathbf{x}, \mathbf{y})$$

Cat = (0, 0, 0, 1, 0, 0, …)

Supervised Learning

$$\vec{x},\ \vec{y} \sim P(\mathrm{x}, \mathrm{y})$$

これはペンです。　　This is a pen.

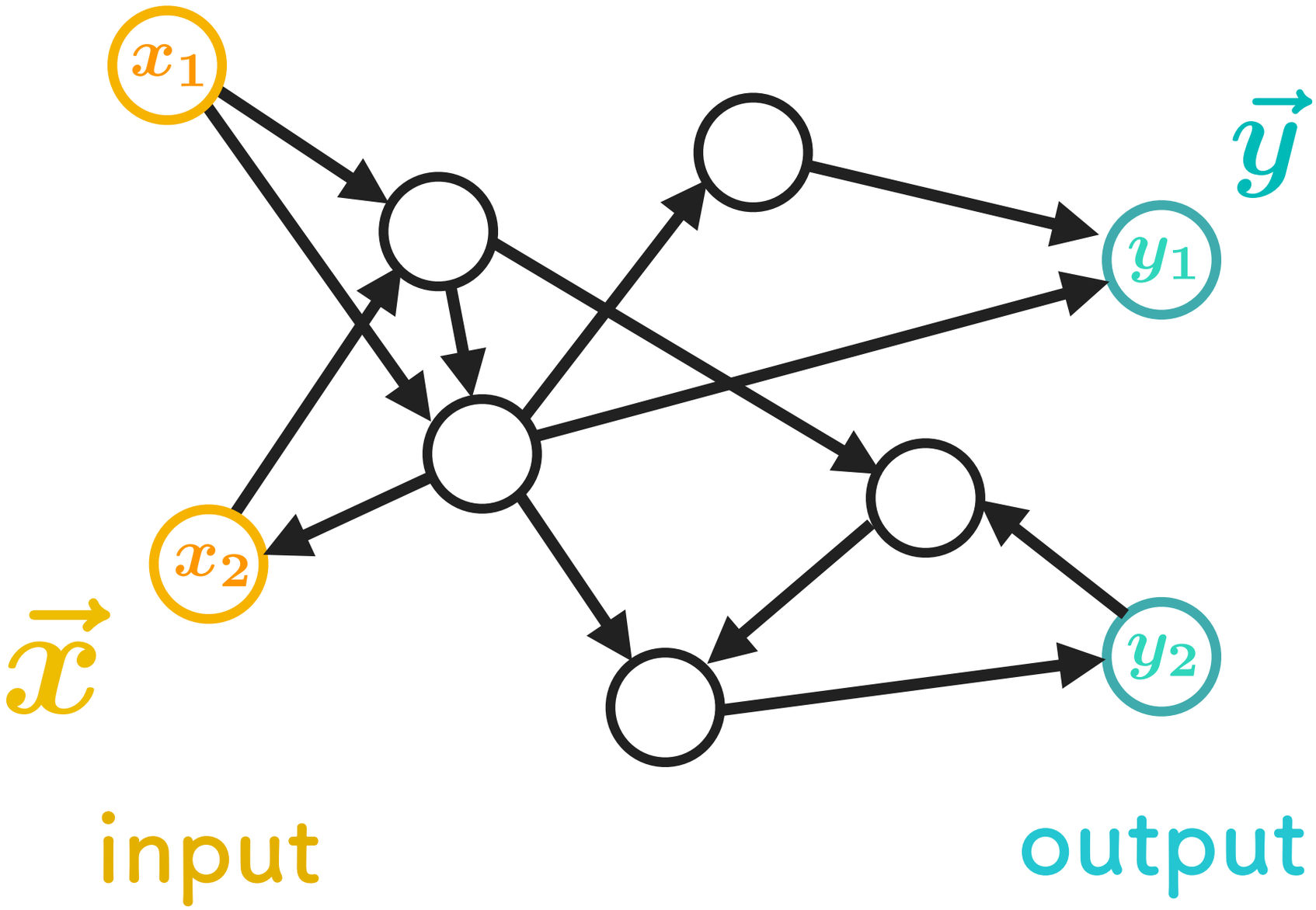
Supervised Learning

$$\vec{x},\ \vec{y} \sim P(\mathrm{x}, \mathrm{y})$$

出力変数を予測する(xからyへの決定論的法則性を推定)

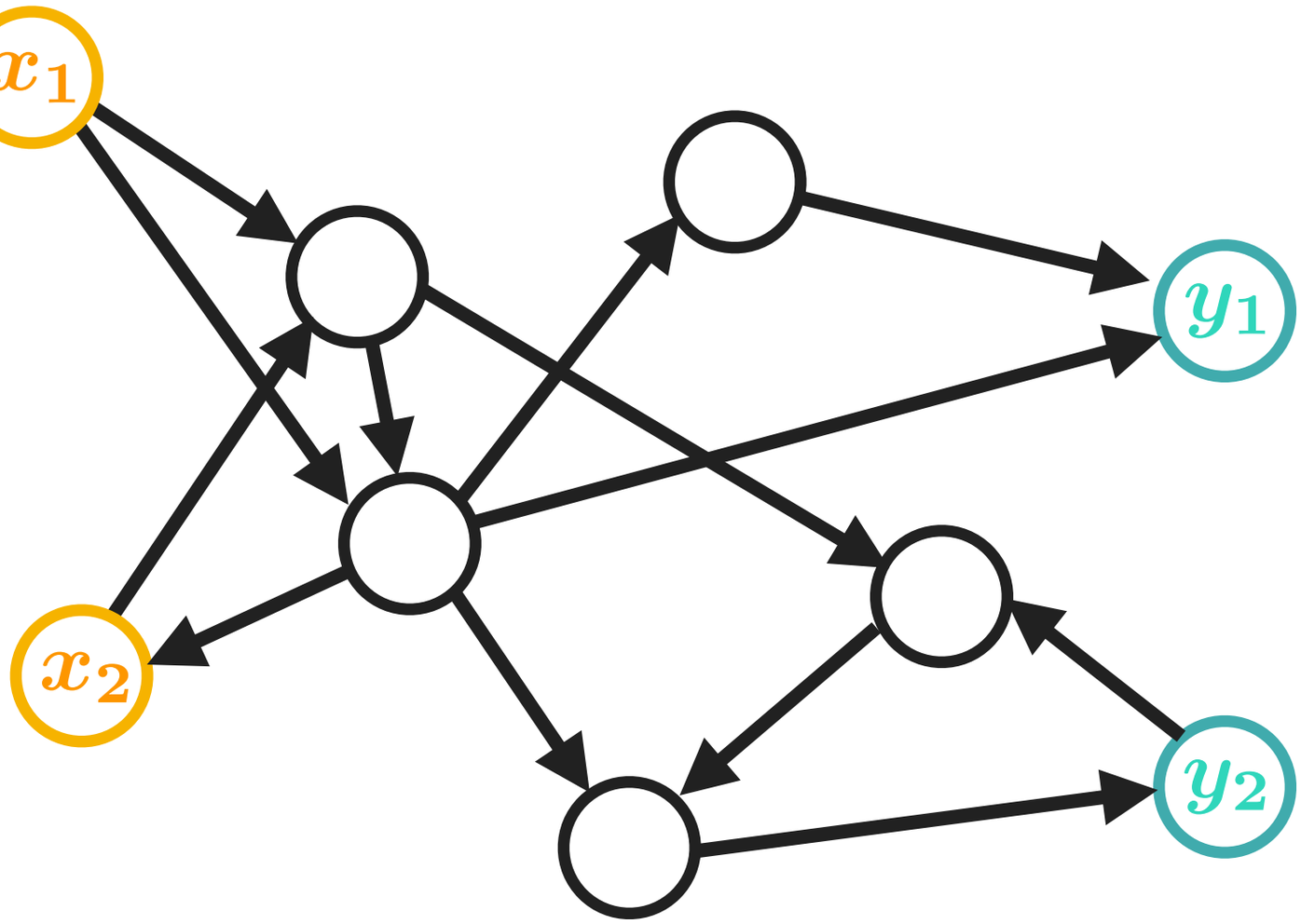
多層ニューラルネットワーク

Model = Directed Graph

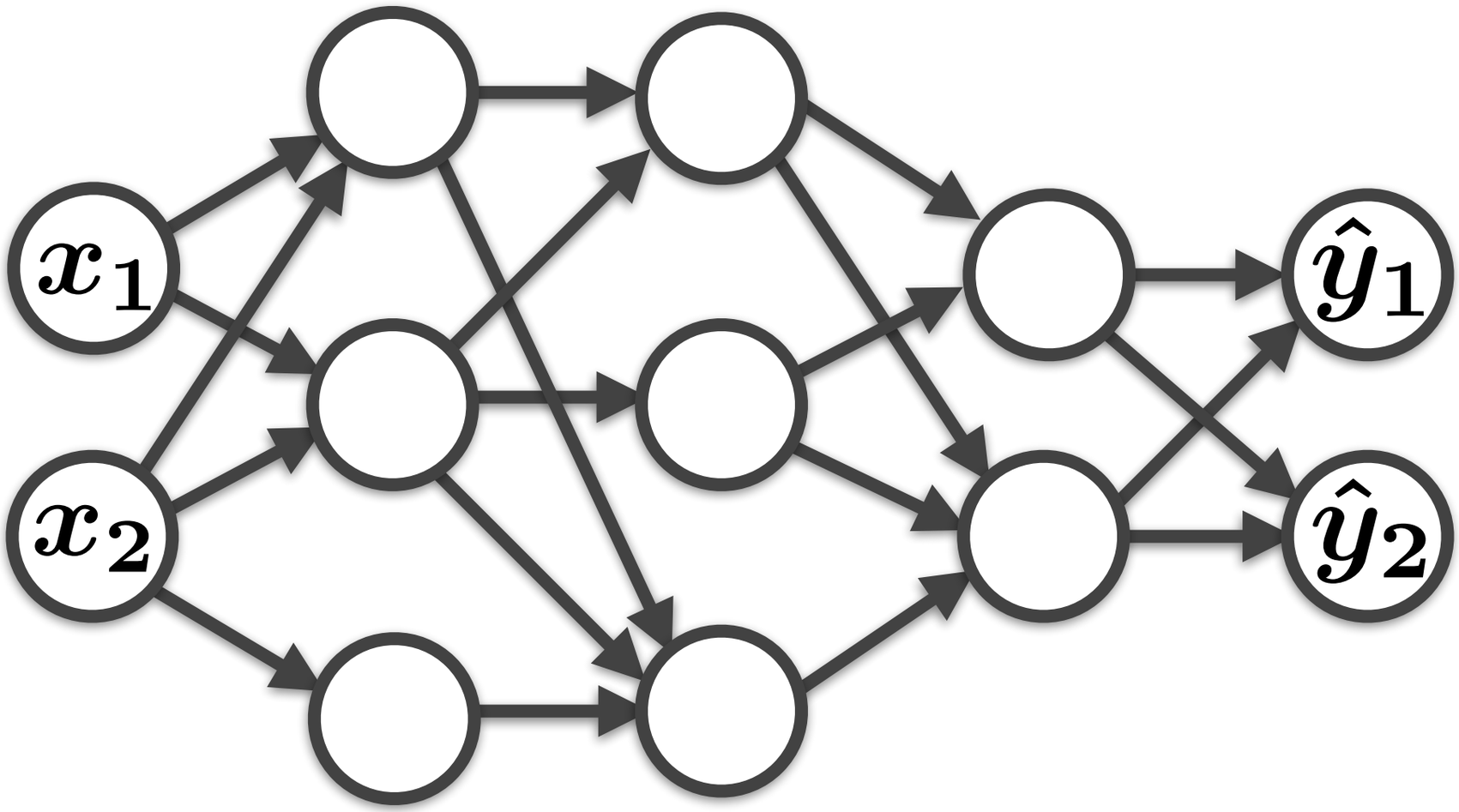


多層ニューラルネットワーク

Model = Directed Graph ← prior (我々の事前知識)



多層ニューラルネットワーク/深層学習



深層学習の規模感

画像認識(1000クラス分類)のケース

・層数：10～1000

・パラメータ数(以下VGG16の例)：138M

これより減らすのが
トレンドだった

・積和演算回数(MACs)：15.5G

深層学習は何をしているか？

多層のネットワークにして結局何が特別なのか？
(極めて高い性能が実現できることは経験的にOK)

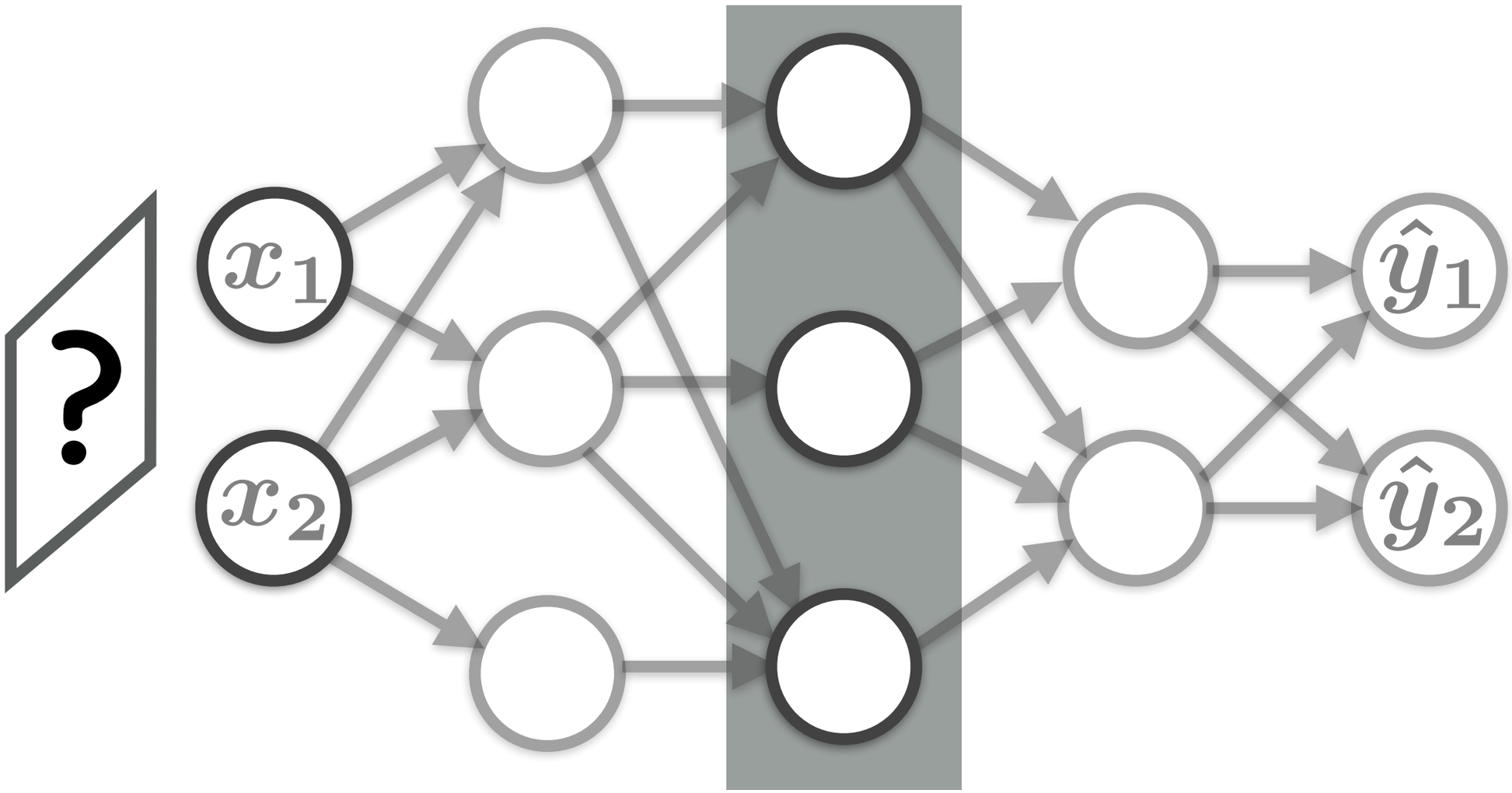
深層学習は何をしているか？

多層のネットワークにして結局何が特別なのか？
(極めて高い性能が実現できることは経験的にOK)

民間信仰：
入力から不要な情報を少しづつ取り除いて、本質だけを最後に取り出す？（くりこみ？）→ No！

VGG16

中間層の信号だけをみて、どれだけ入力信号が再構成できるか？（どれだけ入力情報が残っているか？）



VGG16：第1畳み込み層からの再構成

VGG16：第2畳み込み層からの再構成

VGG16：第3畳み込み層からの再構成

VGG16：第4畳み込み層からの再構成

VGG16：第5畳み込み層からの再構成

VGG16：第6畳み込み層からの再構成

VGG16：第7畳み込み層からの再構成

VGG16：第8畳み込み層からの再構成

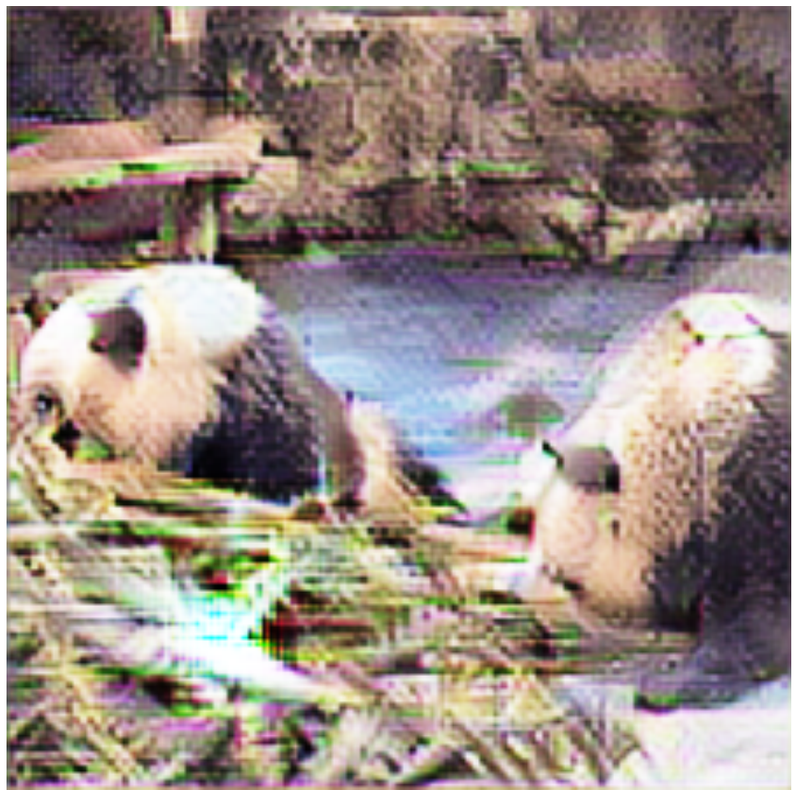
VGG16：第9畳み込み層からの再構成

VGG16：第10畳み込み層からの再構成

VGG16：第11畳み込み層からの再構成

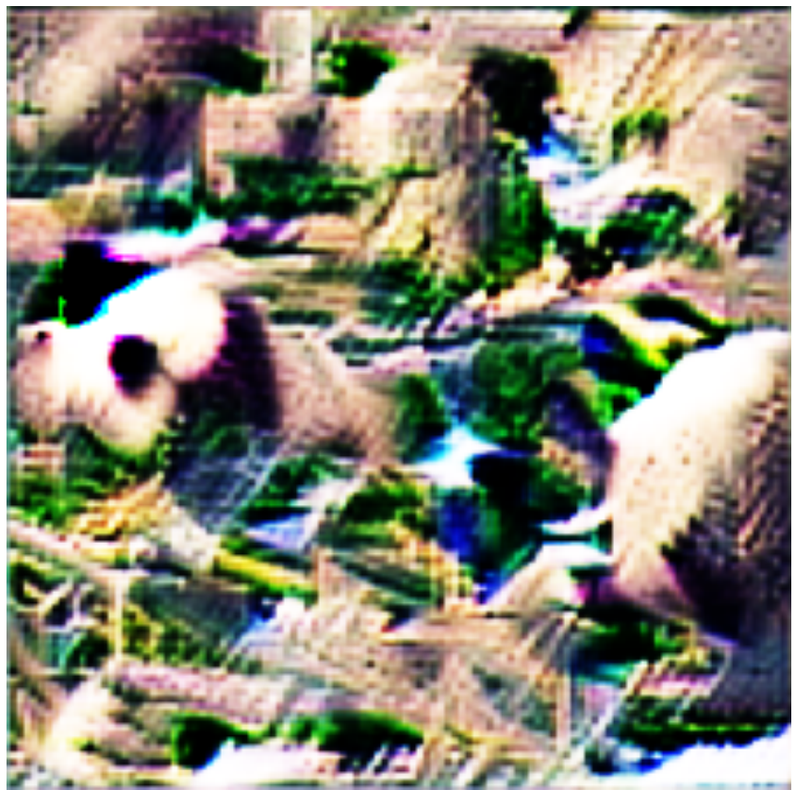
VGG16：第12畳み込み層からの再構成

VGG16：第13畳み込み層からの再構成



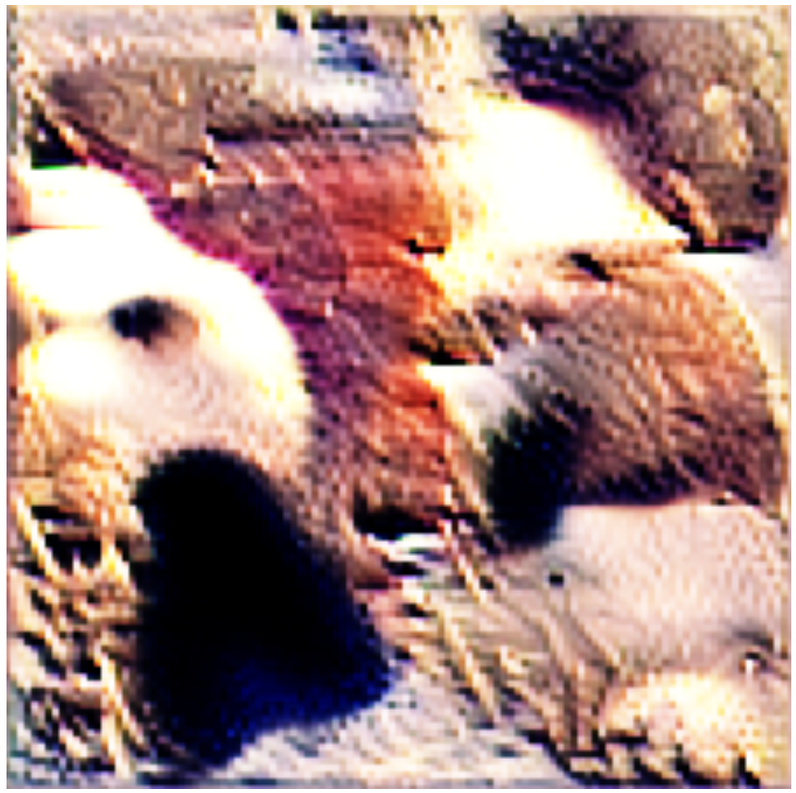
＊再構成困難のため、結果はuniqueではない

VGG16：第14畳み込み層からの再構成



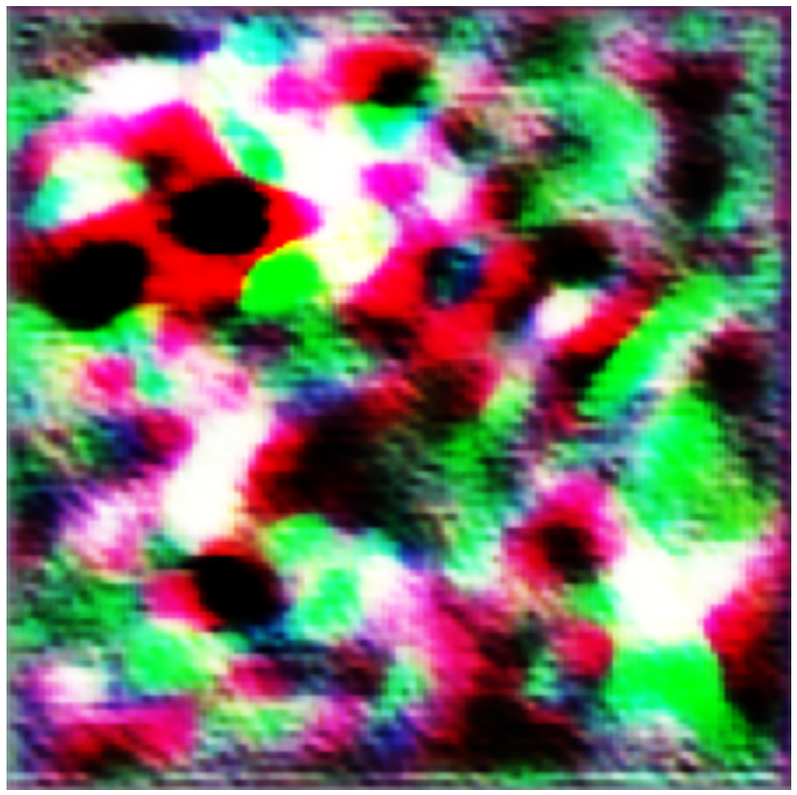
＊再構成困難のため、結果はuniqueではない

VGG16：第15畳み込み層からの再構成



＊再構成困難のため、結果はuniqueではない

VGG16：第16畳み込み層からの再構成



＊再構成困難のため、結果はuniqueではない

深層学習は何をしているか？

後半で急に画像の再構成が困難になる

→ 層の大半は情報を捨てない

深層学習は何をしているか？

後半で急に画像の再構成が困難になる

→ 層の大半は情報を捨てない

深層学習≒多ステップにわたる情報の整理整頓

良い情報表現を作るため、単に情報を並び替えて整理しているだけ（ではないか）。

深層学習は何をしているか？

いずれにせよ、なぜ深層学習が特別なのかは未だによくわかっていない！

今日はその一側面の話をします。

1. 予備知識

勾配降下法による最適化

Supervised Learning

観測データセット

$$(x_1, y_1)$$
$$(x_2, y_2)$$
$$\vdots$$
$$(x_N, y_N)$$

Supervised Learning

観測データセット

⟶ 予測の誤差

Supervised Learning

観測データセット

⟶ 予測の誤差

E.g. Mean Square Error

$$E(w) = \frac{1}{N} \sum_{n=1}^{N} \left(\hat{y}(x_n; w) - y_n \right)^2$$

$$\longrightarrow \quad w^* = \mathrm{argmin}_w E(w)$$

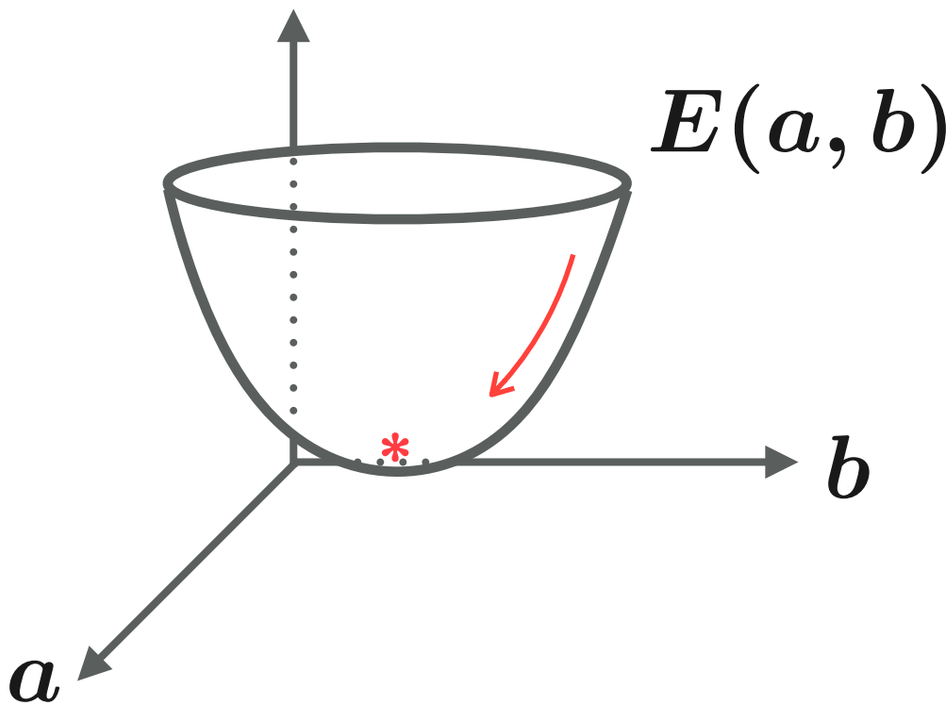
Supervised Learning = 最小化

データの**学習**

=

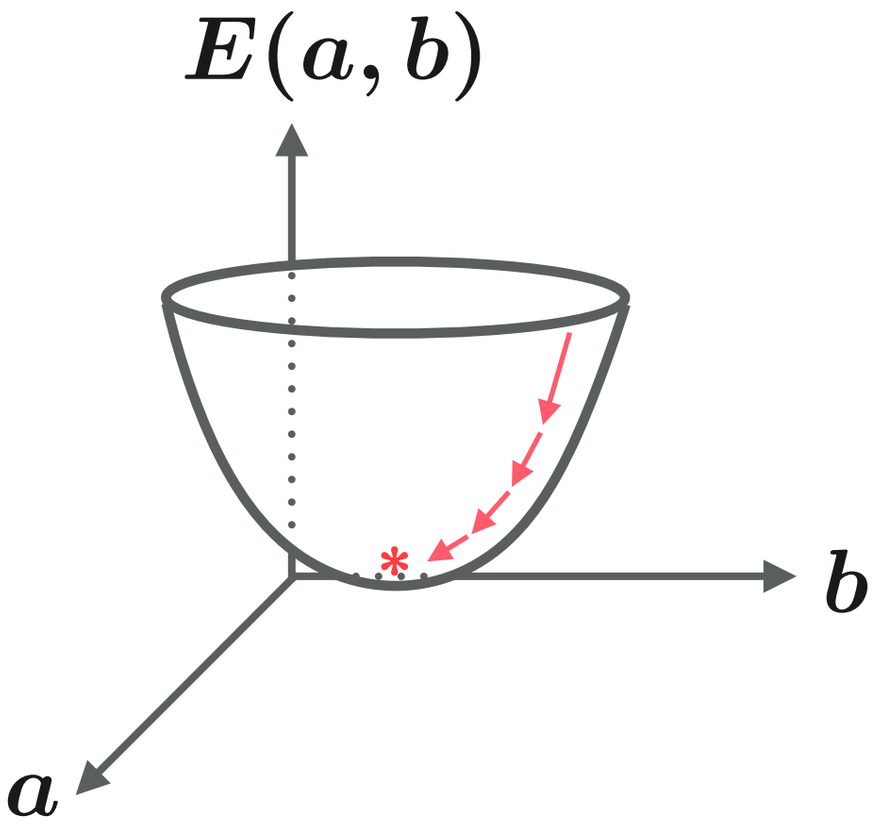
最適化(最小化)問題

$$a^*, b^* = \mathrm{argmin}\, E(a, b)$$



勾配降下法

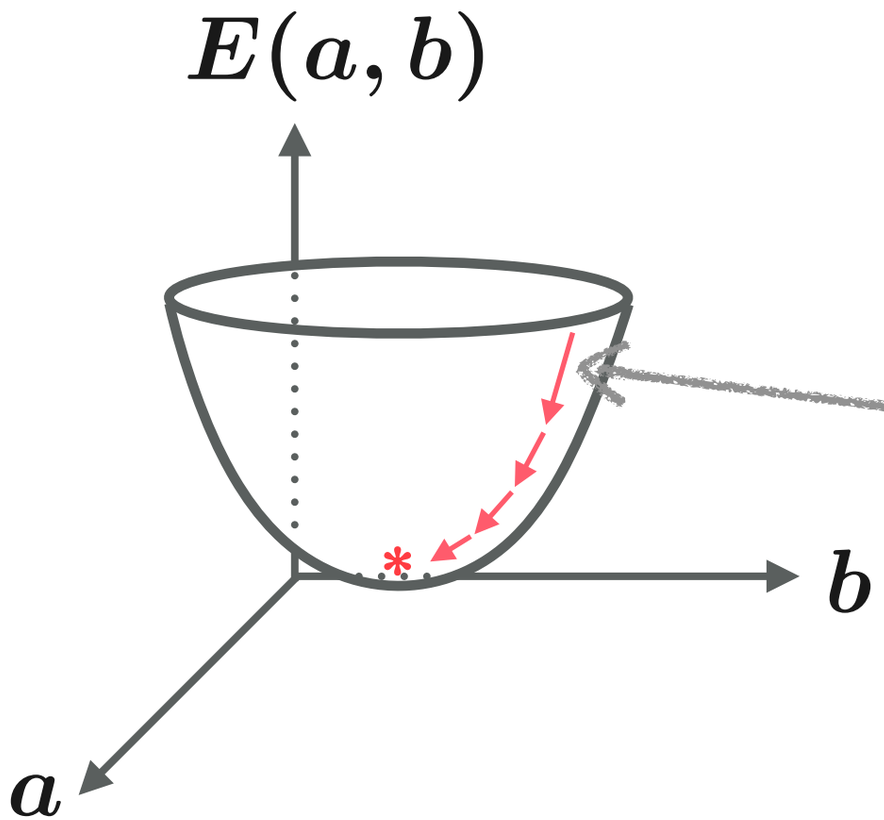
最適化(最小化)問題を数値的に解く



坂道を転がすように動かしていけば、やがて底で止まる

勾配降下法

最適化(最小化)問題を数値的に解く



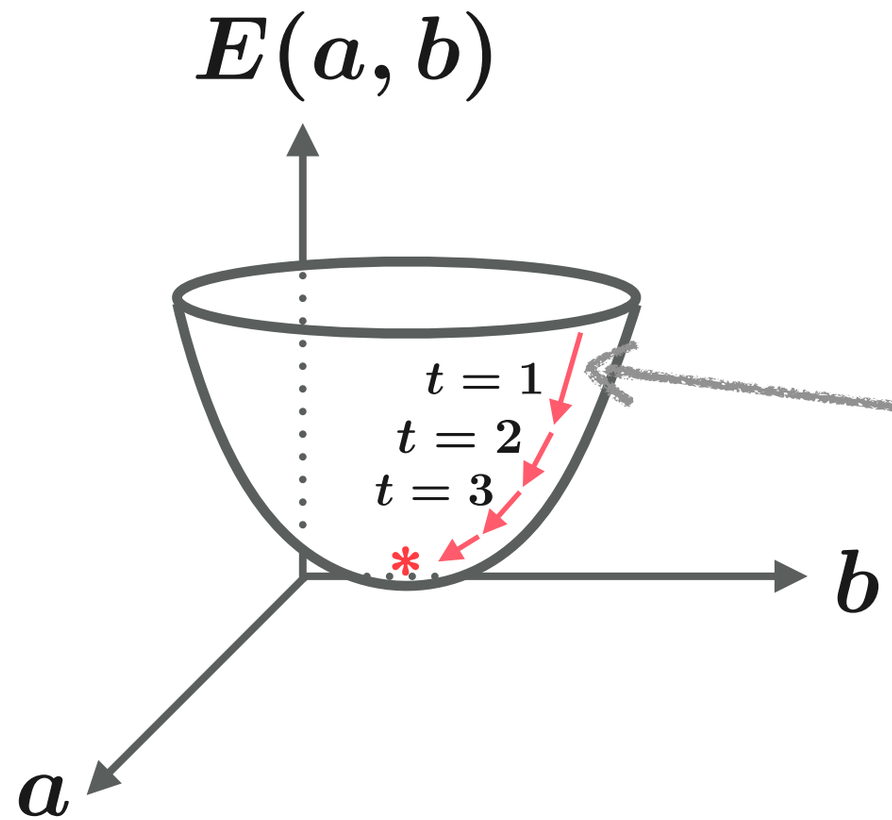
坂道を転がすように動かしていけば、やがて底で止まる

$$-\begin{pmatrix} \frac{\partial E}{\partial a} \\ \frac{\partial E}{\partial b} \end{pmatrix}$$

勾配 gradient

勾配降下法

最適化(最小化)問題を数値的に解く



坂道を転がすように動かしていけば、やがて底で止まる

$$-\begin{pmatrix} \frac{\partial E}{\partial a} \\ \frac{\partial E}{\partial b} \end{pmatrix}$$

勾配降下
gradient descent

学習率

$$a^{t+1} = a^t - 0.01\frac{\partial E(a^t, b^t)}{\partial a}$$

$$b^{t+1} = b^t - 0.01\frac{\partial E(a^t, b^t)}{\partial b}$$

ニューラルネットワークのオンライン学習

ミニバッチ学習

$$E(a,b) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{y}(x_n : a,b) - y_n\right)^2$$

ミニバッチ学習

$$E(a,b) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{y}(x_n : a,b) - y_n\right)^2$$

勾配法を使うときは、各時刻tで毎回わざわざ全部のデータに関して平均を取らなくても良い。ランダムに一部だけデータを取ってきて(部分集合=ミニバッチ)、それに関して誤差関数を計算

ミニバッチ学習

$$E(a,b) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{y}(x_n : a,b) - y_n\right)^2$$

$$E^t(a,b) = \frac{1}{M}\sum_{m=1}^{M}\left(\hat{y}(x_m : a,b) - y_m\right)^2$$

毎時刻、ランダムに部分集合を選び直す

$$\{(x_m, y_m)\}_{m=1}^{M} \subset \{(x_n, y_n)\}_{n=1}^{N}$$

オンライン学習

極端な場合は、ミニバッチがデータ点一個だけ（M=1）

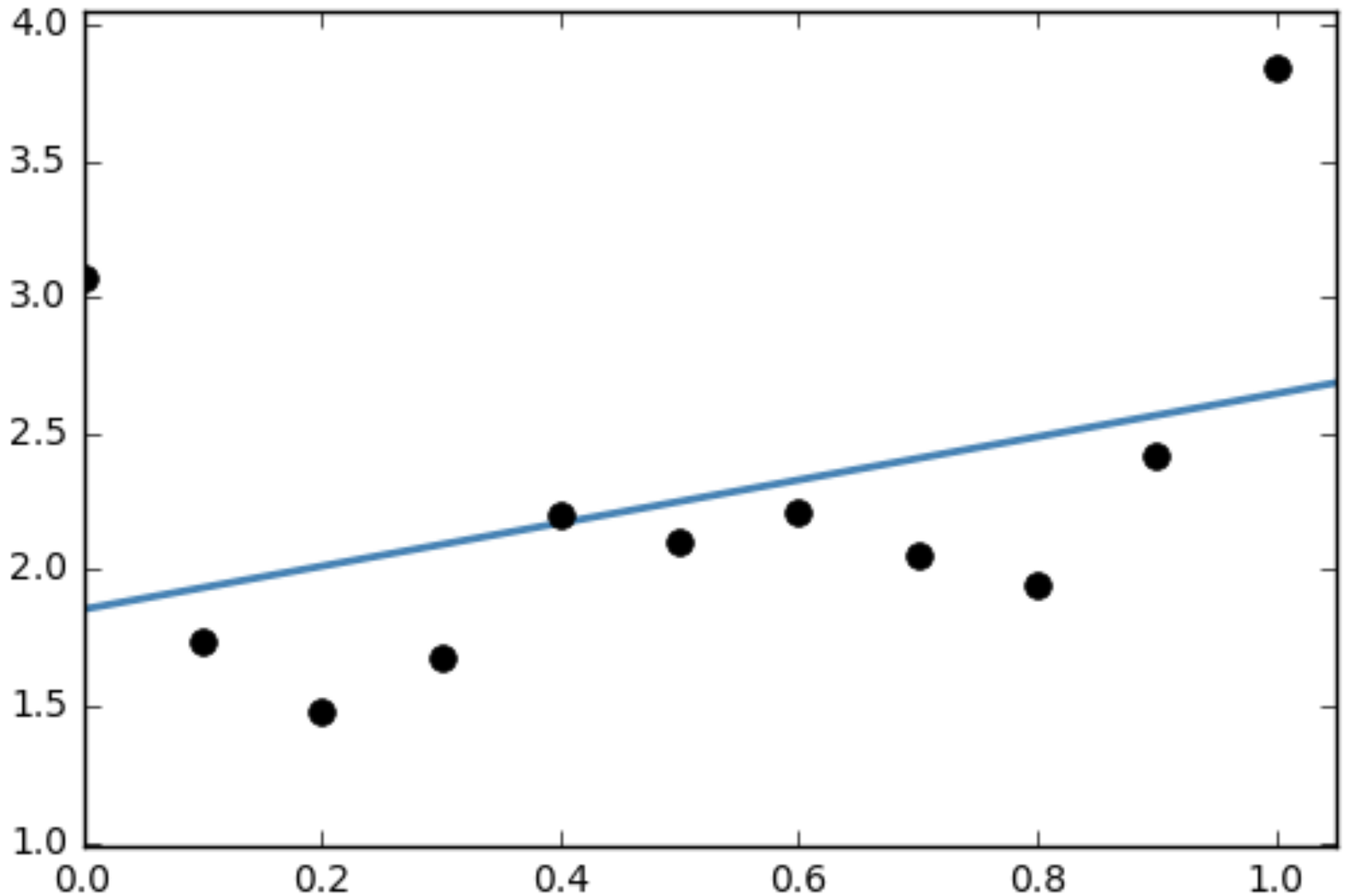
$$E^t(a,b) = \frac{1}{M}\sum_{m=1}^{M}\left(\hat{y}(x_m : a,b) - y_m\right)^2$$

毎時刻、ランダムに部分集合を選び直す

$$\{(x_m, y_m)\}_{m=1}^{M} \subset \{(x_n, y_n)\}_{n=1}^{N}$$

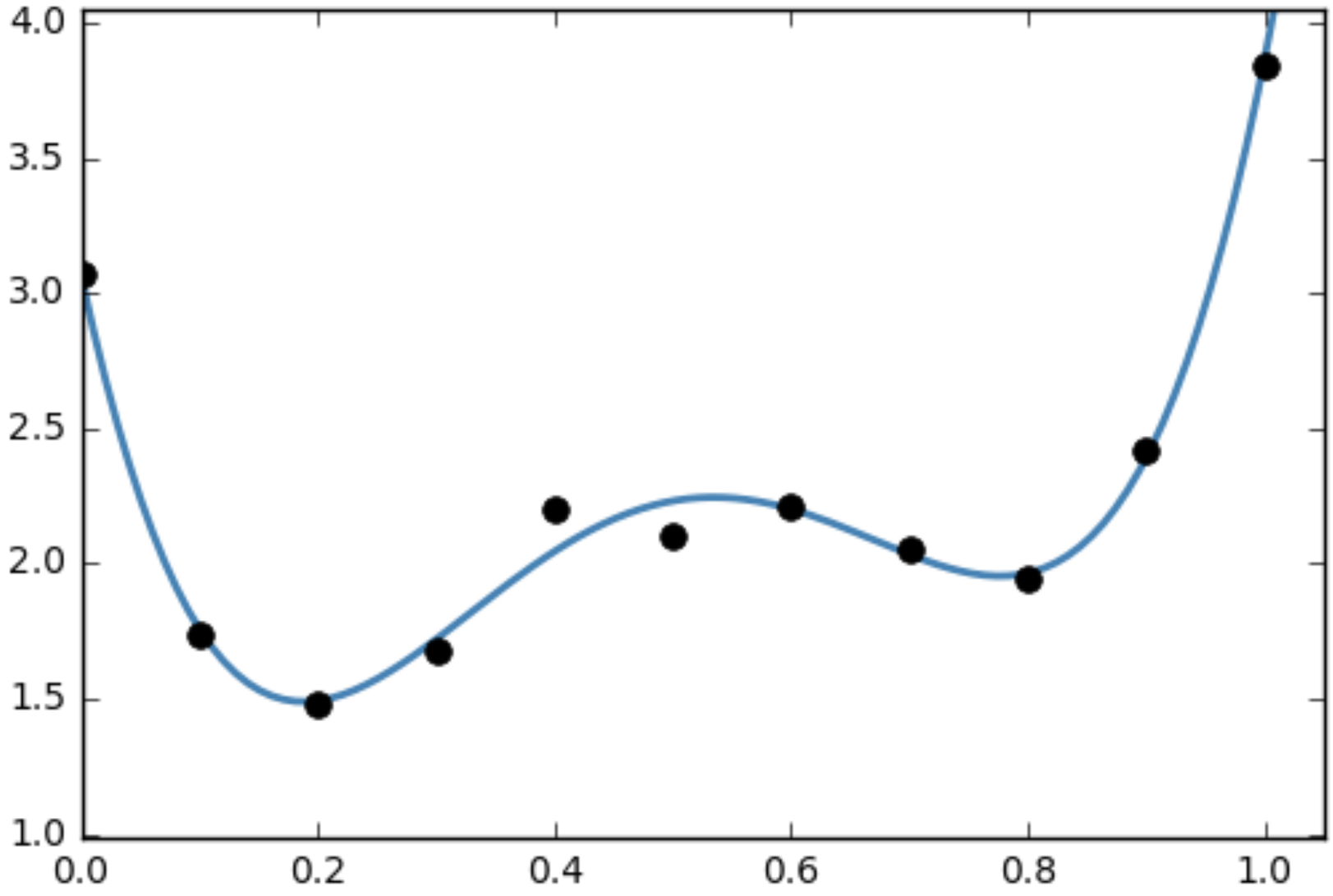
2. 機械学習と汎化

最適化問題を解くだけではダメ



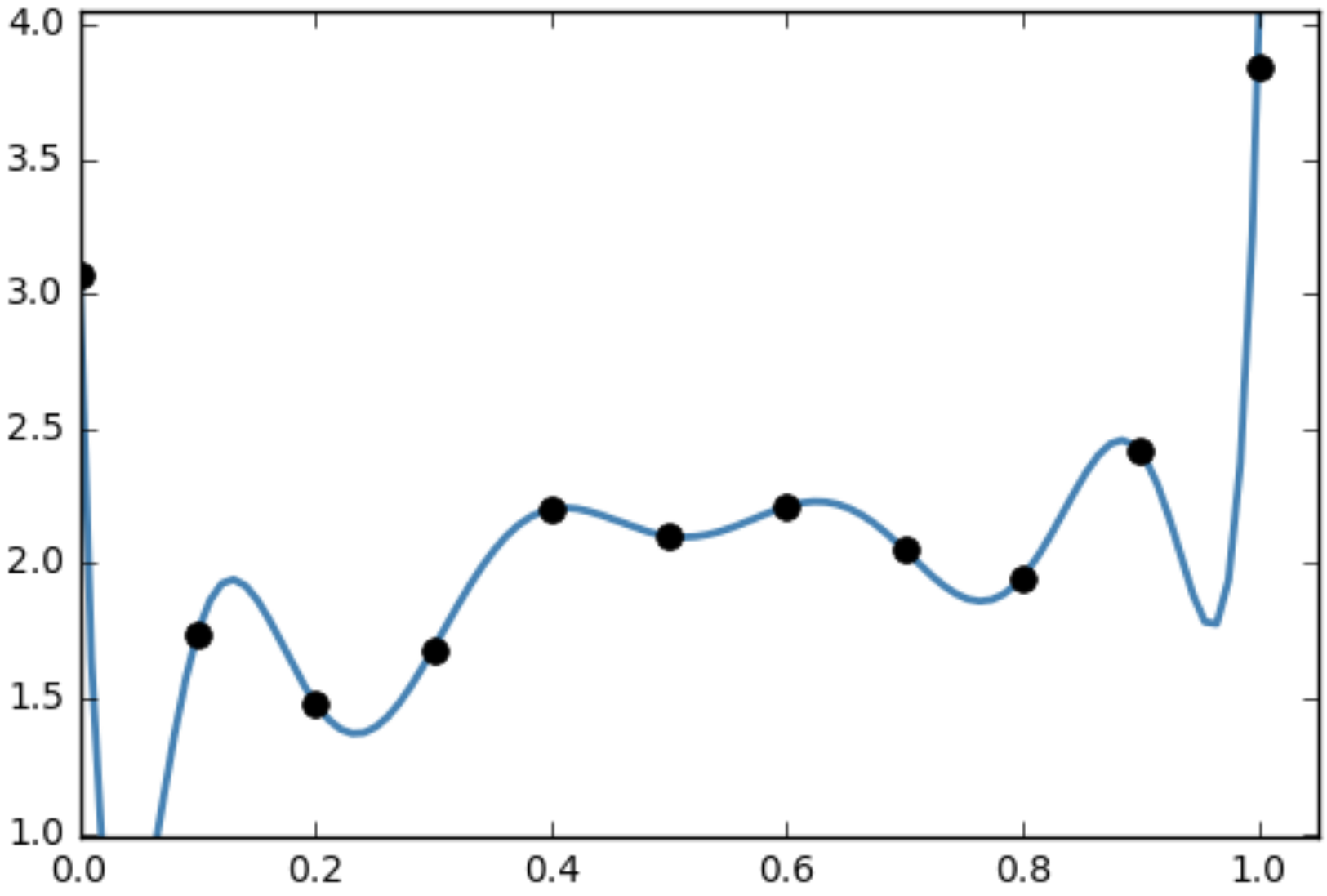
未学習 underfitting

最適化問題を解くだけではダメ



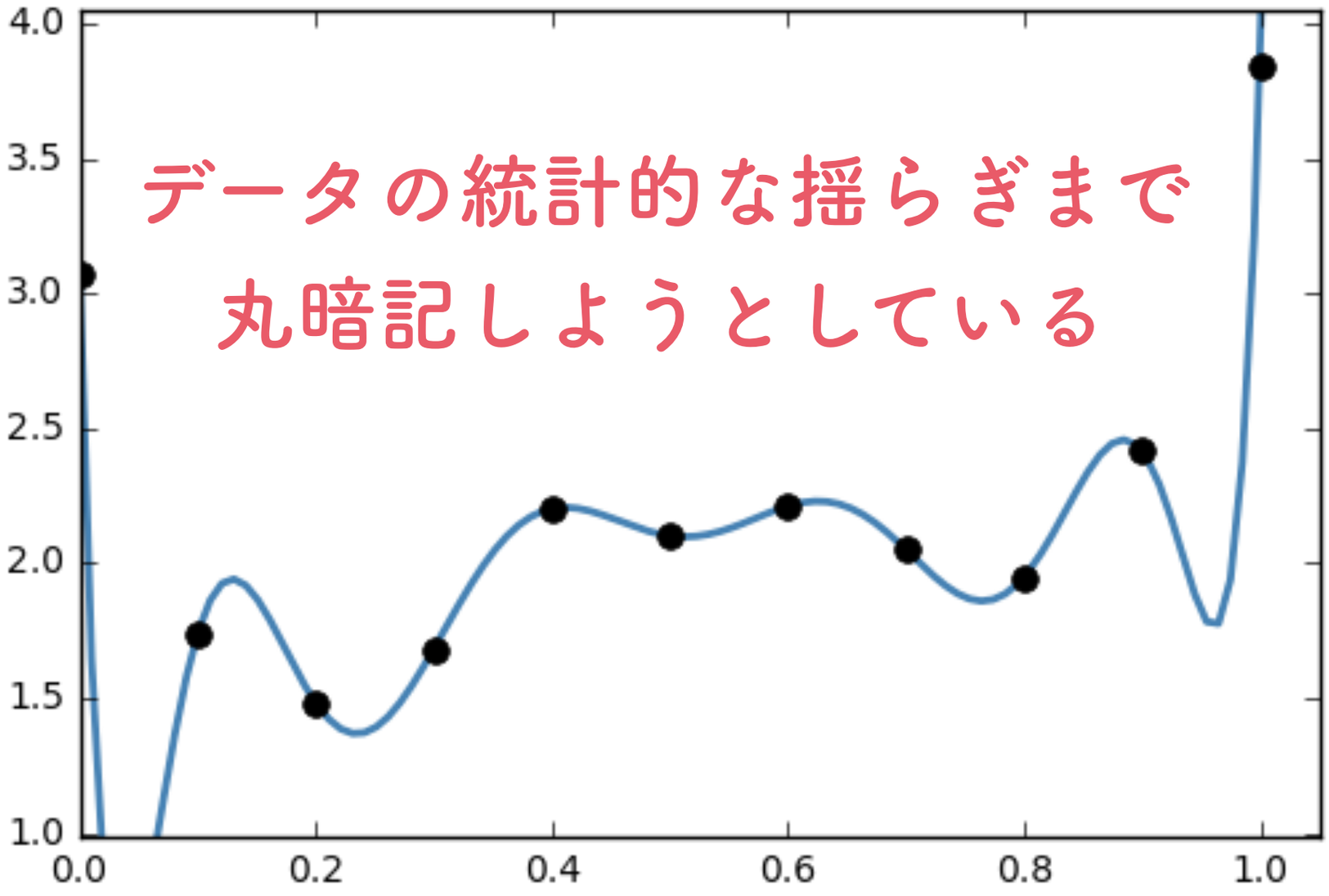
程よい（汎化）

最適化問題を解くだけではダメ



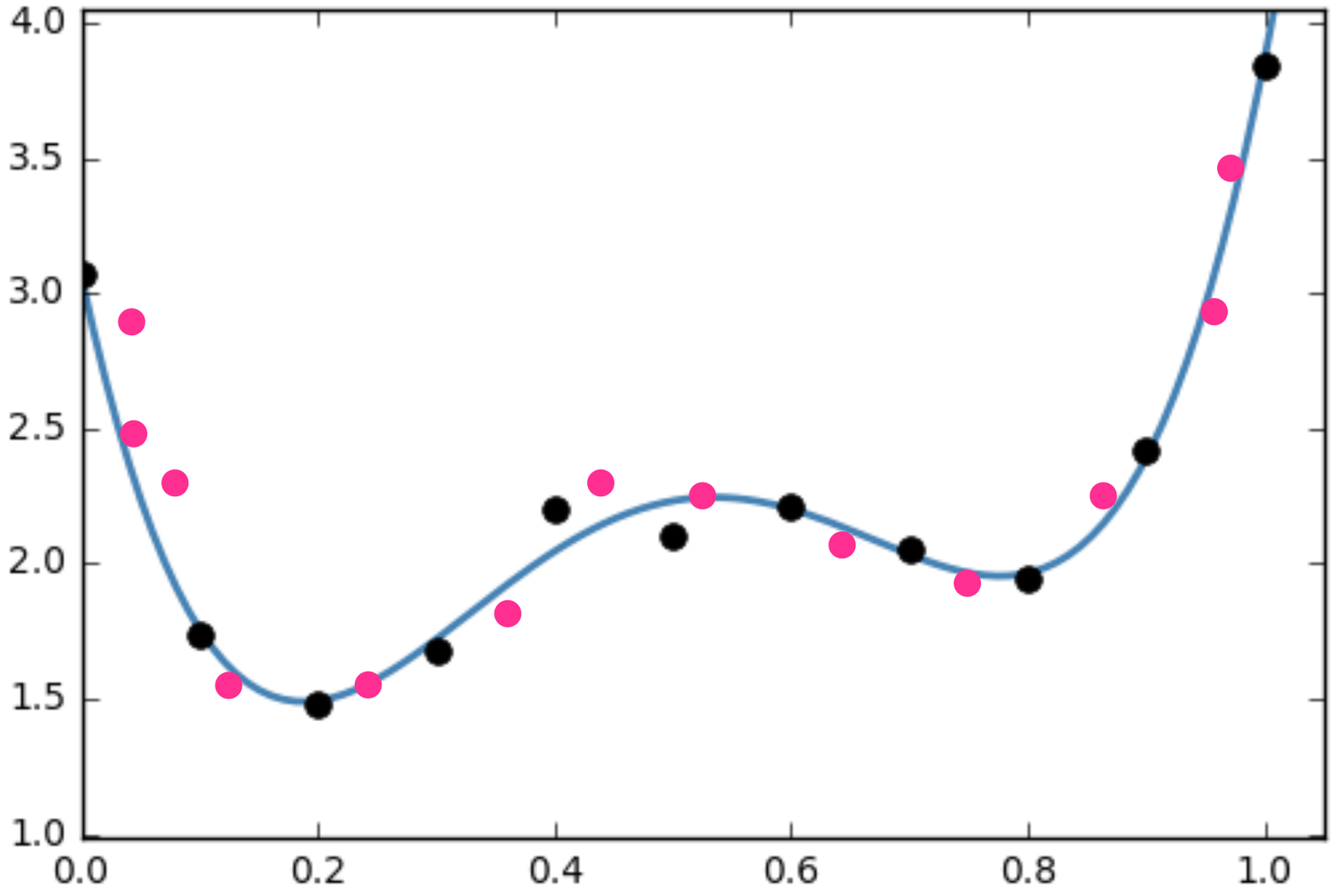
過学習 overfitting

最適化問題を解くだけではダメ



過学習 overfitting

汎化 generalization



ロバストなパターンだけ捉えて、ノイズは無視する。それにより
未知のデータに対してまで当てはまる予測モデルが得られる

あらわな正則化 explicit regulazitation

仮説から、学習の過程で余分な自由度が
削減されるようにする（などの）手法

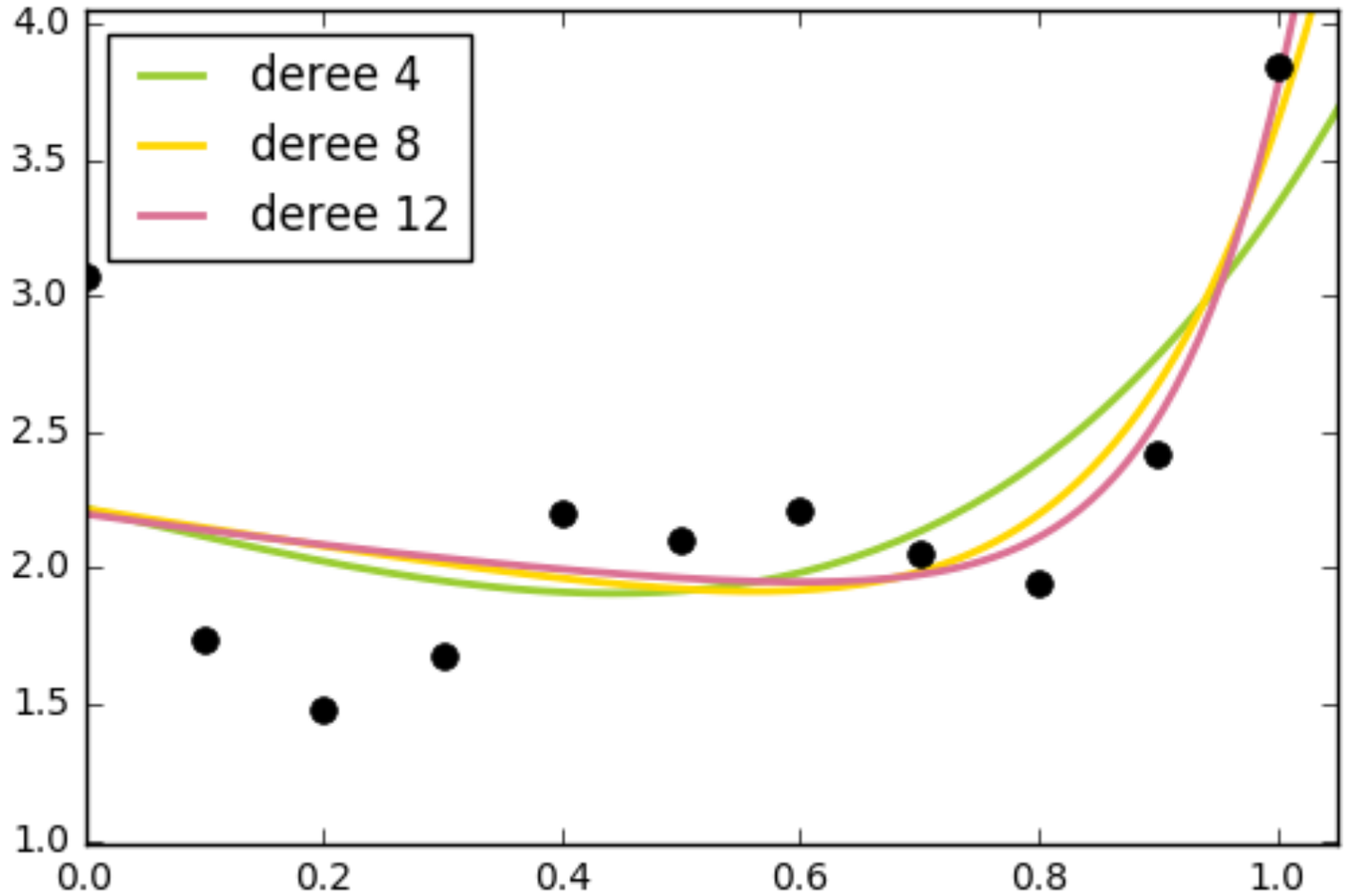
学習アルゴリズム（～最小化する誤差関数）
を変更して、過剰な自由度を抑制する

あらわな正則化 explicit regulazitation

$$E(W) \to E(W) + \lambda \sum \left(W_{ij}^{(\ell)}\right)^2$$

重み減衰（仮説にpreferenceを置く）

あらわな正則化 explicit regulazitation



めっちゃ強くかけると、ほとんど仮説の次数によらなくなる

深層学習の実態

過剰なパラメータを持つDNN

＋

適切な正則化(⇒モデル選択)

＋

大きなデータセット（豊かな内容を持つ問題）

＋

マルチコア上での並列化

↓

桁外れに高いパターン認識精度

深層学習の実態

過剰なパラメータを持つDNN

＋

適切な正則化(⊇ モデル選択)

＋

大きなデータセット（豊かな内容を持つ問題）

＋

マルチコア上での並列化

↓

桁外れに高いパターン認識精度

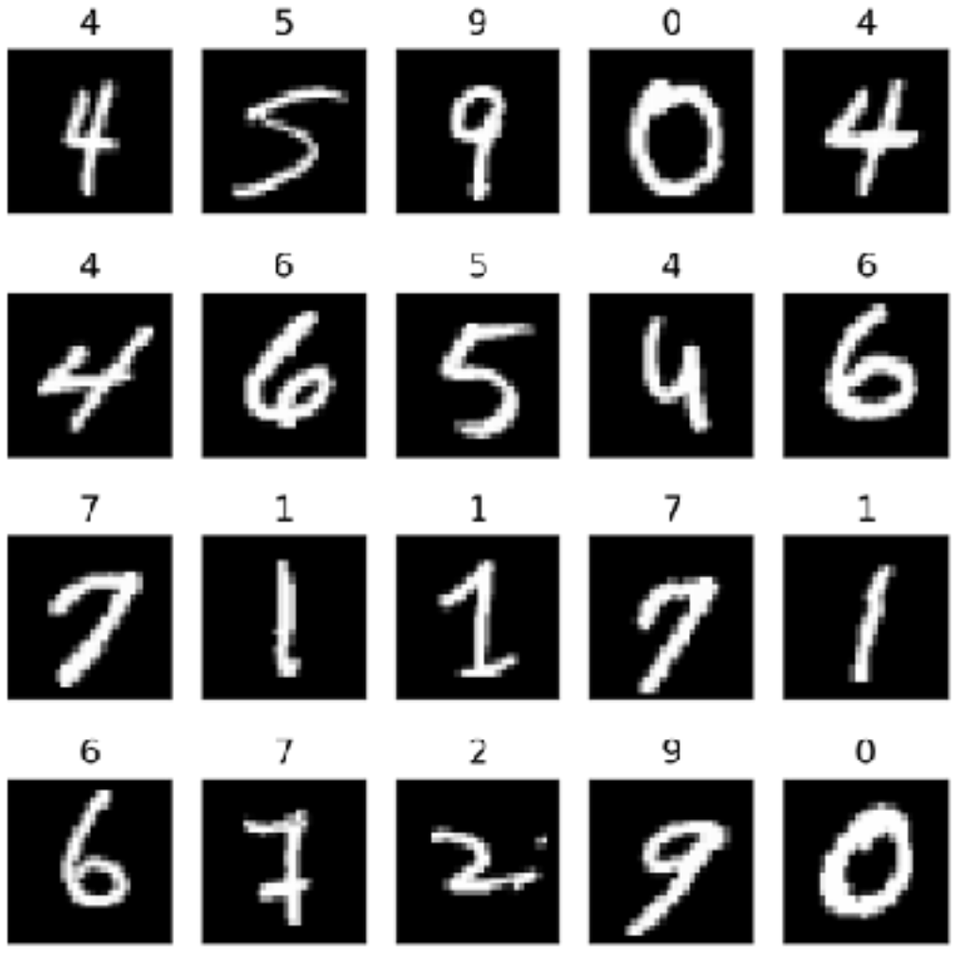
3. 深層学習の汎化は謎である

汎化の謎

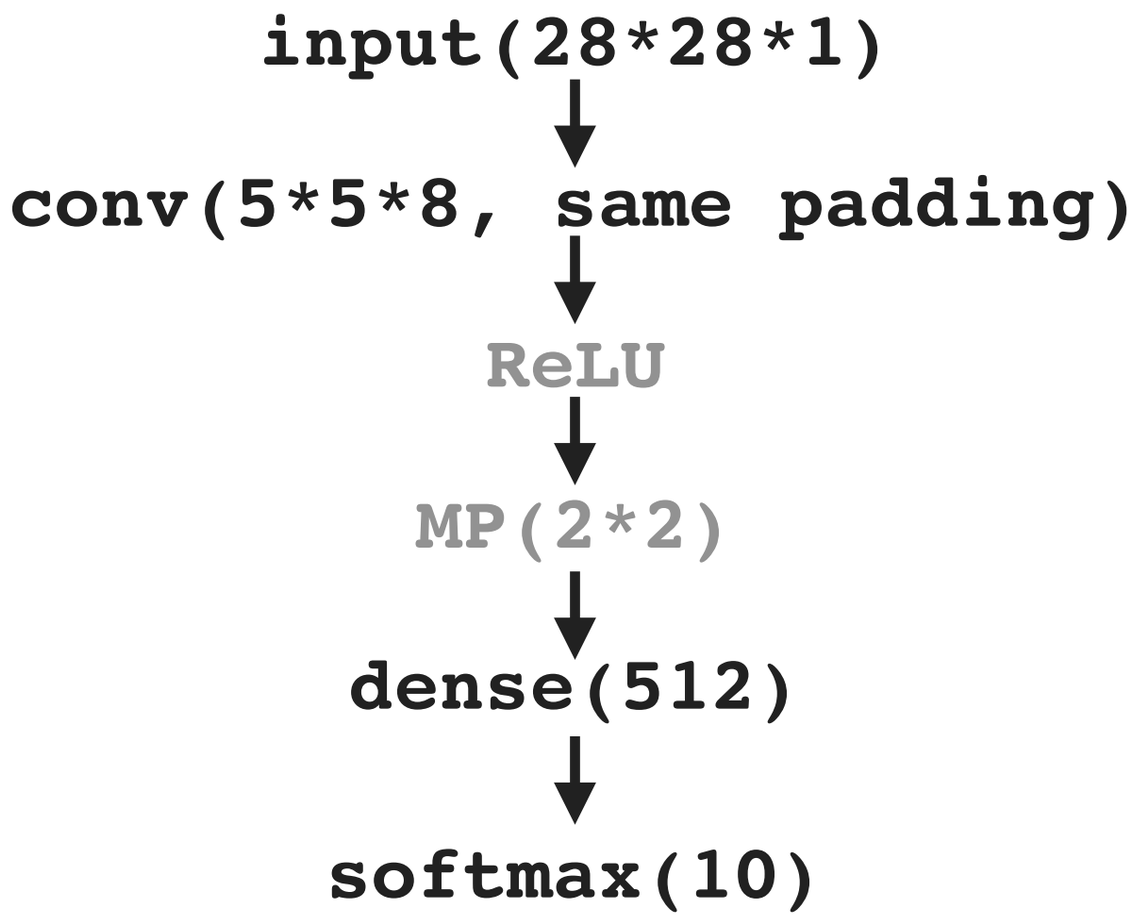
データの内容に比べて、深層ネットワークには過剰なまでに大量なパラメータがあります。普通はこのような状況は丸暗記や過剰適合を招き、学習が失敗するのですが、深層学習はなぜかうまくいきます。正則化のおかげだと昔は思われてきたのですが…

過剰なパラメータ：丸暗記と汎化

MNISTデータセット



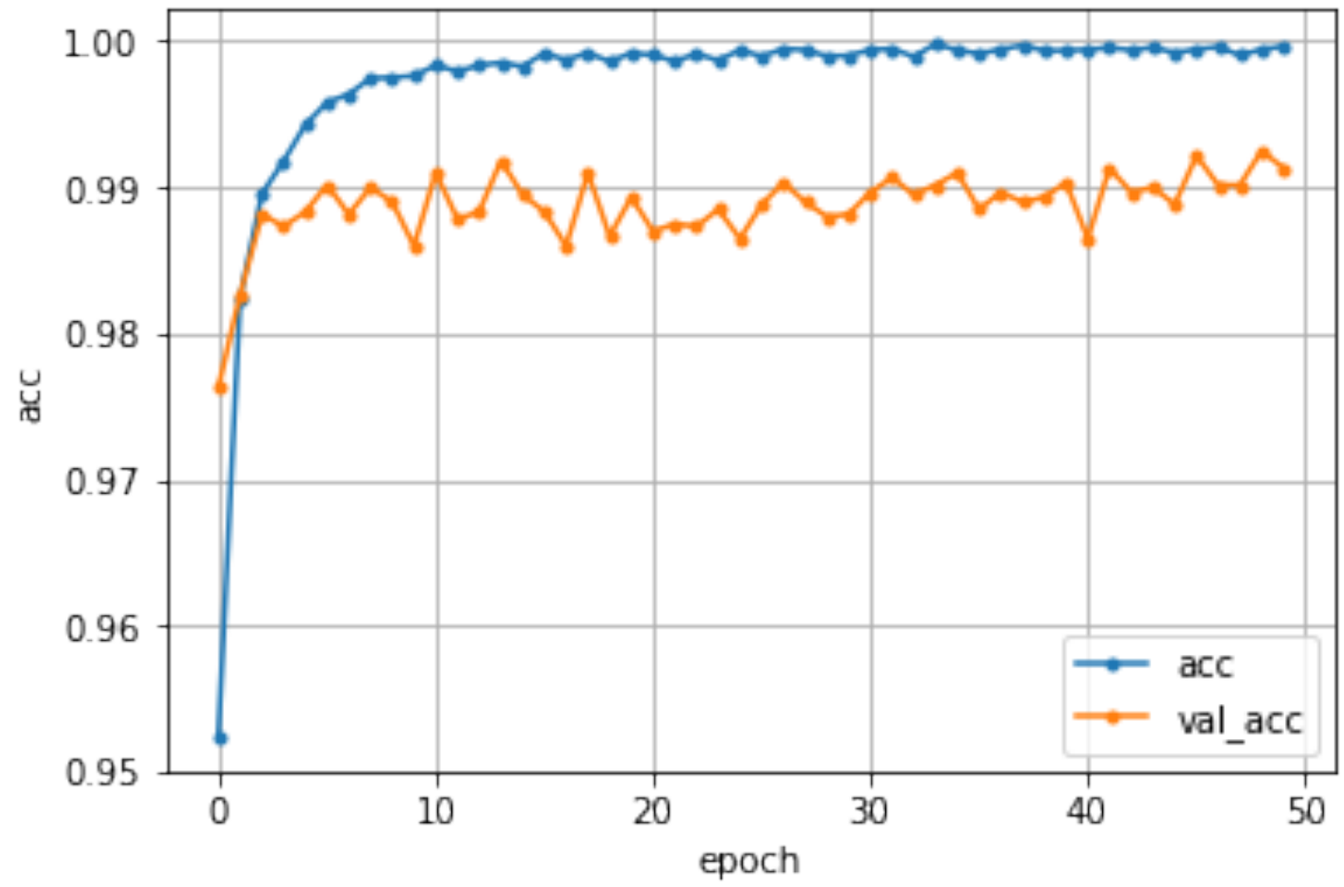
過剰なパラメータ：丸暗記と汎化



でかいモデルですが訓練してみます（80万パラメータ）。

ミニバッチサイズ 32、adam optimizer、正則化なし

過剰なパラメータ：丸暗記と汎化



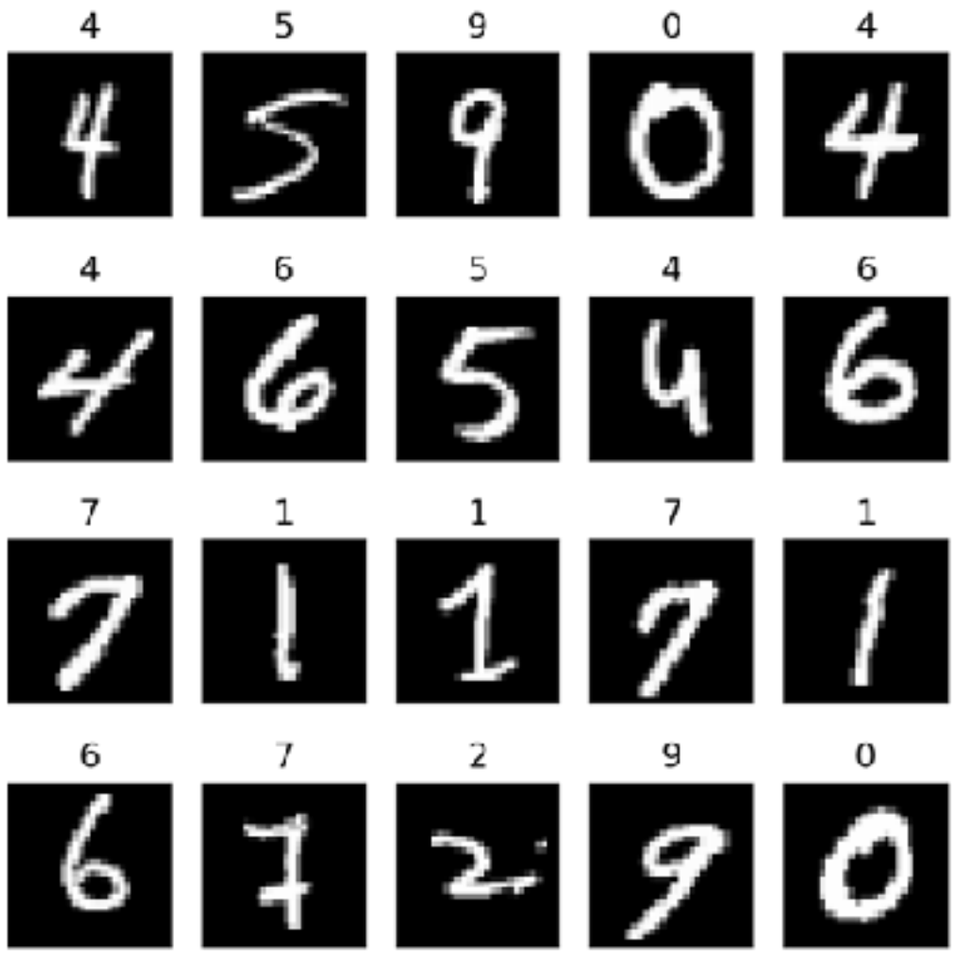
良い性能に収束（正解率99%以上）

普通のチュートリアルレベルの話

過剰なパラメータ：丸暗記と汎化

本当に正則化の下で自由度を削減しながら、普遍的なパターンだけを抽出しているのだろうか？

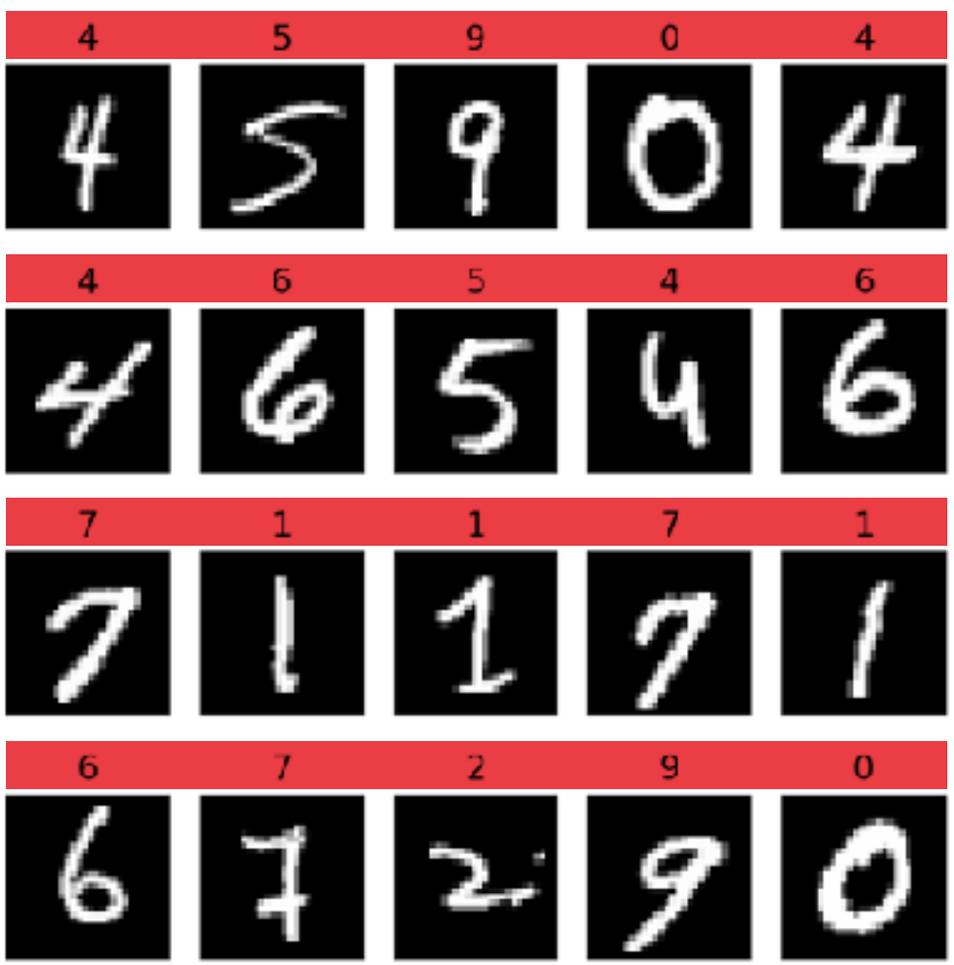
実験：ランダムラベルデータセットによる学習



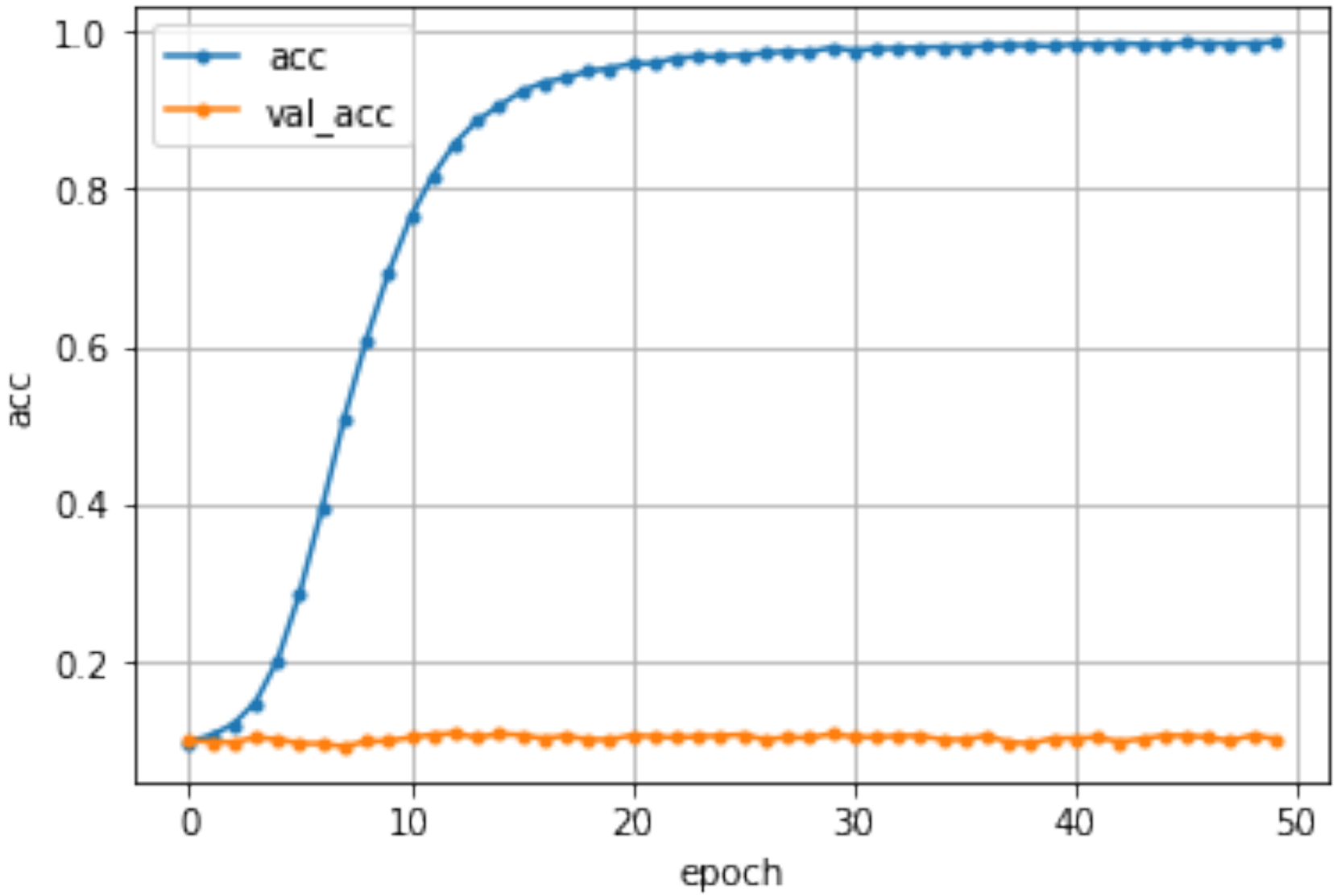
過剰なパラメータ：丸暗記と汎化

本当に正則化の下で自由度を削減しながら、普遍的なパターンだけを抽出しているのだろうか？

実験：ランダムラベルデータセットによる学習



過剰なパラメータ：丸暗記と汎化



memorization

by chance 0.1

ランダムデータも極めてよく学習できている！！！

過剰なパラメータ：丸暗記と汎化

過剰なパラメータにもかかわらず、汎化するのは謎

explicitな正則化で有効自由度を下げて、ロバストなパターンを抽出するのが汎化と考えられてきた

しかし実は同じ設定で、パターンの無いものまで学習できてしまう。つまり丸暗記できてしまうsetupなのに汎化する！！

過剰なパラメータ：丸暗記と汎化

[Zhang et al., 2016]など

モデルの良さ(平均的な汎化能力)は、ネットワーク構造のデザインでほとんど決まっている。

不思議：丸暗記できるのに、パターンがあるときはしない！

認識困難な事例については暗記。これは無駄な学習になるので、正則化はこれ(single directions)を防ぐ。

モデル選択の効果 [Zhang et al., 2016]

Table 1: The training and test accuracy (in percentage) of various models on the CIFAR10 dataset. Performance with and without data augmentation and weight decay are compared. The results of fitting random labels are also included.

ほとんどモデル選択で性能は決まる

model	# params	random crop	weight decay	train accuracy	test accuracy
Inception	1,649,402	yes	yes	100.0	89.05
		yes	no	100.0	89.31
		no	yes	100.0	86.03
		no	no	100.0	85.75
(fitting random labels)		no	no	100.0	9.78
Inception w/o BatchNorm	1,649,402	no	yes	100.0	83.00
		no	no	100.0	82.00
(fitting random labels)		no	no	100.0	10.12
Alexnet	1,387,786	yes	yes	99.90	81.22
		yes	no	99.82	79.66
		no	yes	100.0	77.36
		no	no	100.0	76.07
(fitting random labels)		no	no	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
(fitting random labels)		no	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
(fitting random labels)		no	no	99.34	10.61

良いモデルでも丸暗記はできてしまう

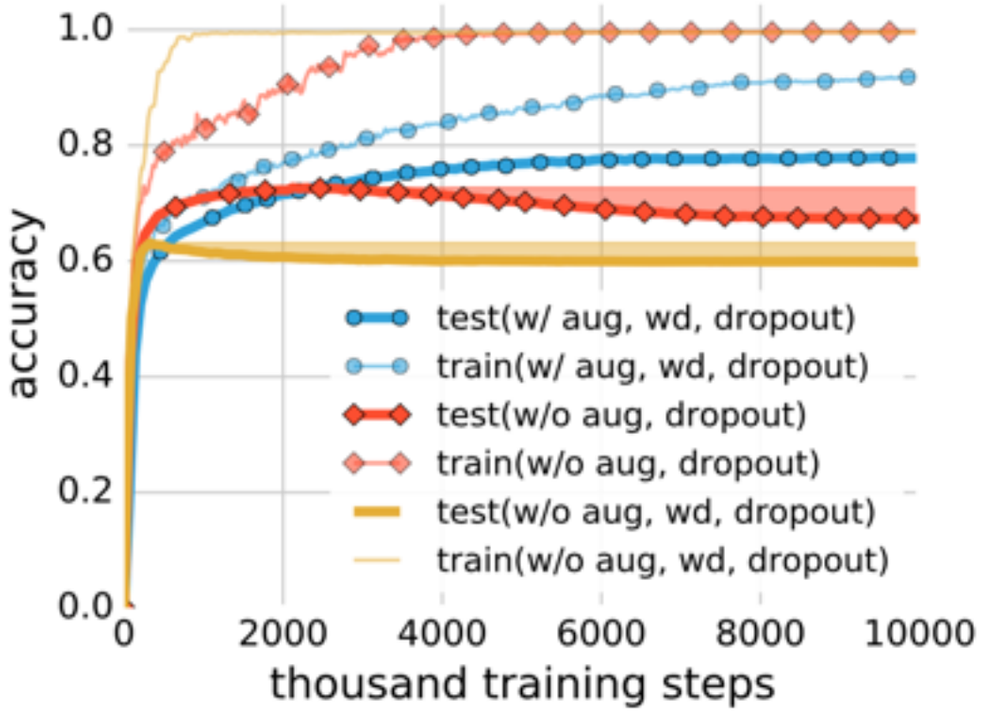
あらわな正則化の効果 [Zhang et al., 2016]

Table 2: The top-1 and top-5 accuracy (in percentage) of the Inception v3 model on the ImageNet dataset. We compare the training and test accuracy with various regularization turned on and off, for both true labels and random labels. The original reported top-5 accuracy of the Alexnet on ILSVRC 2012 is also listed for reference. The numbers in parentheses are the best test accuracy during training, as a reference for potential performance gain of early stopping.

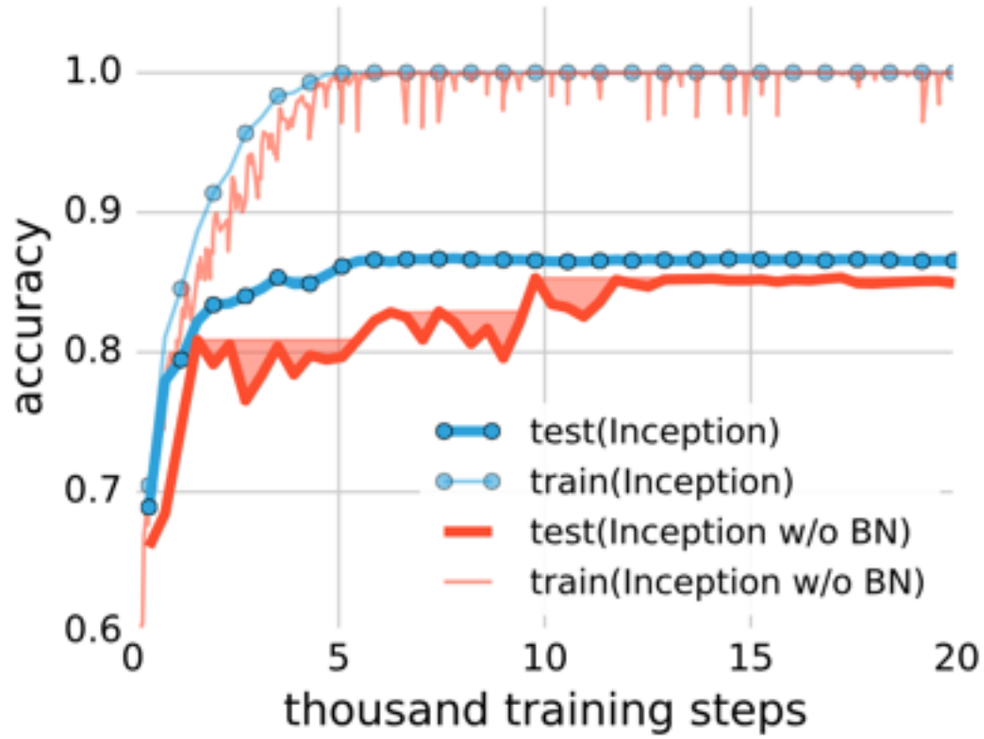
data aug	dropout	weight decay	top-1 train	top-5 train	top-1 test	top-5 test
ImageNet 1000 classes with the original labels						
yes	yes	yes	92.18	99.21	77.84	93.92
yes	no	no	92.33	99.17	72.95	90.43
no	no	yes	90.60	100.0	67.18 (72.57)	86.44 (91.31)
no	no	no	99.53	100.0	59.80 (63.16)	80.38 (84.49)
Alexnet (Krizhevsky et al., 2012)			-	-	-	83.6
ImageNet 1000 classes with random labels						
no	yes	yes	91.18	97.95	0.09	0.49
no	no	yes	87.81	96.15	0.12	0.50
no	no	no	95.20	99.14	0.11	0.56

正則化などは暗記を防いでいる

あらわな正則化の効果 [Zhang et al., 2016]



(a) Inception on ImageNet



(b) Inception on CIFAR10

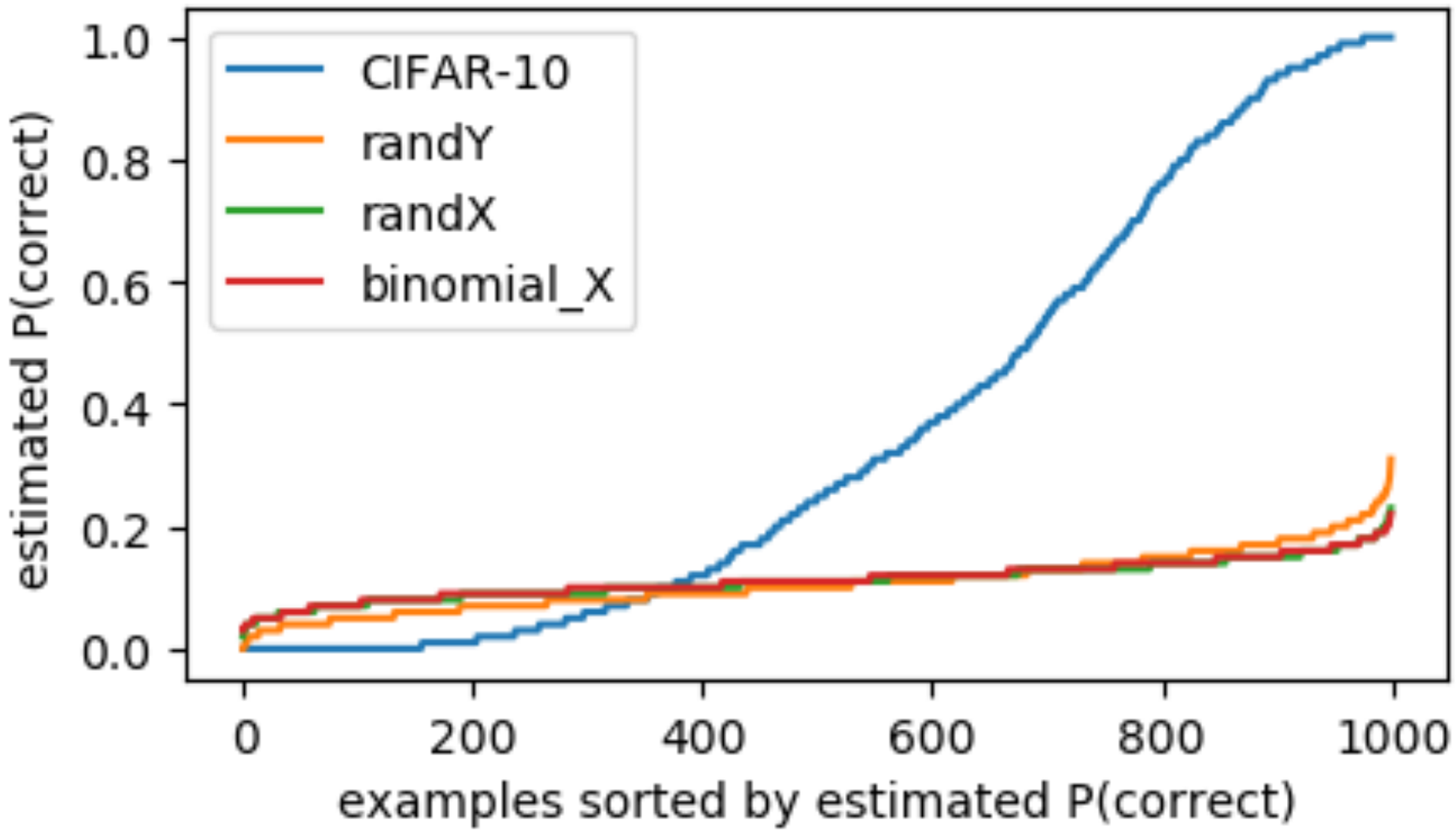
モデル選択（=陰な正則化）が性能の大部分を決めている

SGDが極小値を選別？

ランダムデータでの1エポック後の結果 (6層CNN)

[D.Arpit et al., 2017]

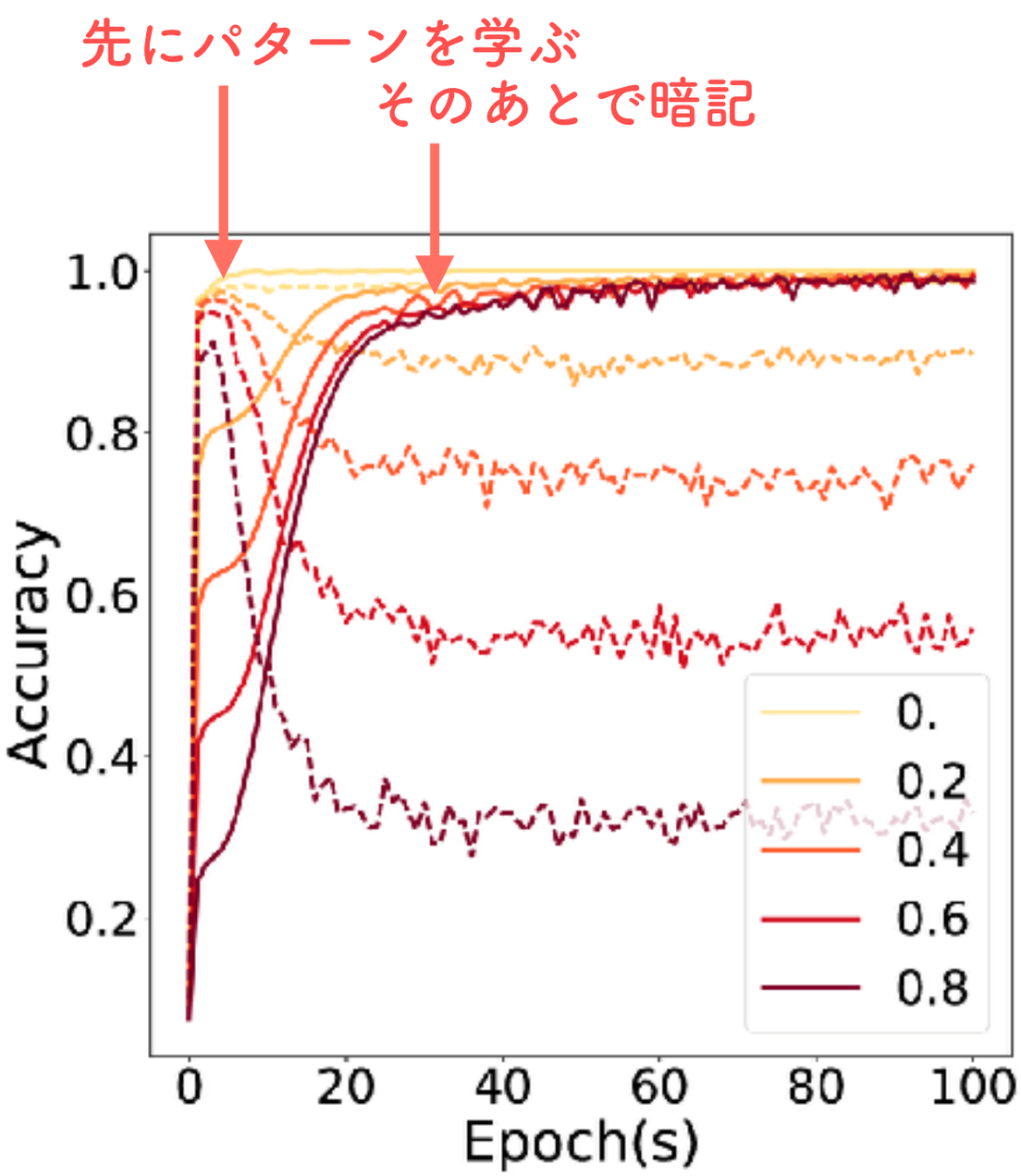
正解率
（100回の実験平均）

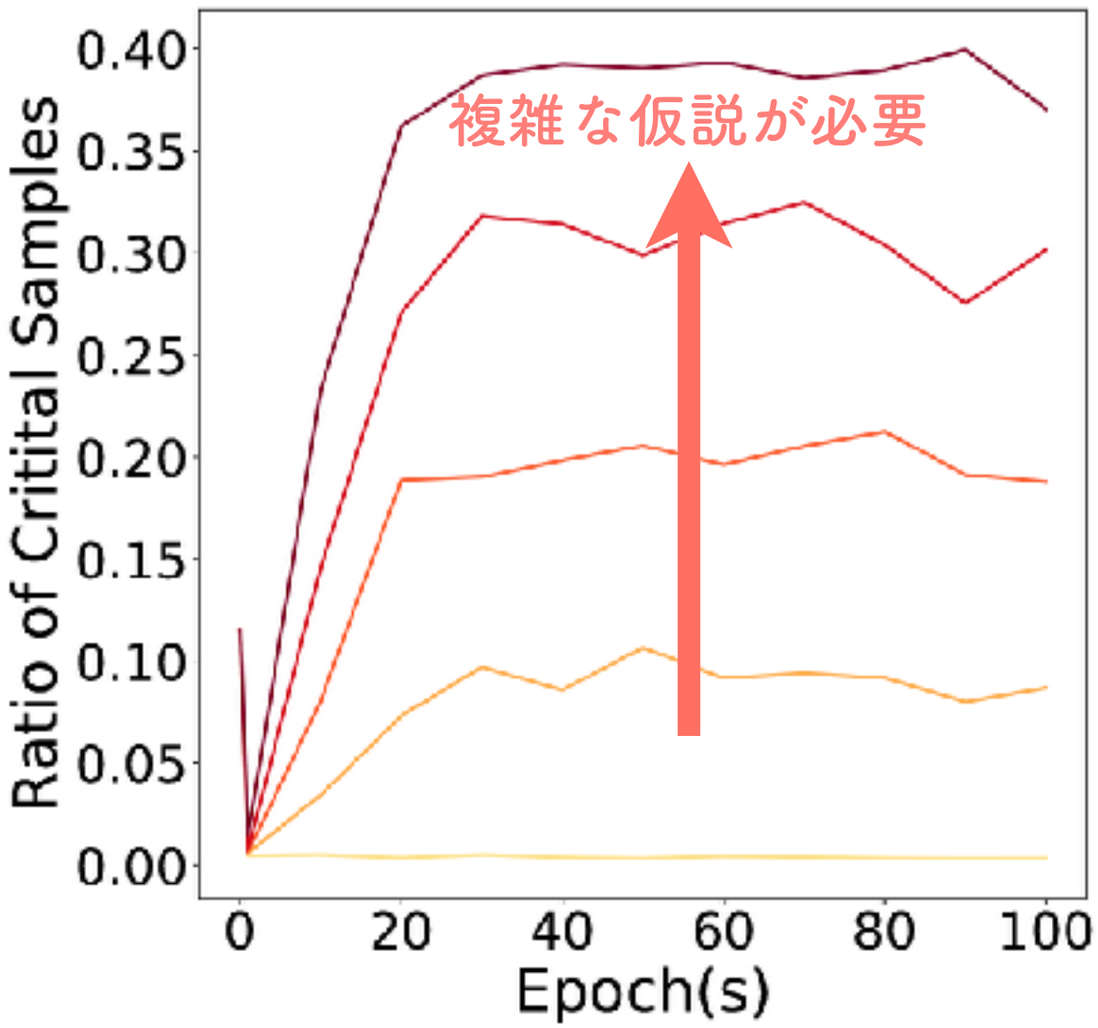


1000個のデータ点（画像）をそれぞれ小さい順に並べた

実データでは、学習後も認識の「容易い画像」と「難しい画像」がある。容易いパターンは学習の初期から他のデータへも汎化。ランダムデータはほぼ同等に「難しい」

critical sample ratio [D.Arpit et al., 2017]





0-80％の割合でMNISTの標的側yをランダムノイズで置き換え、「難しいデータ点」の割合を変えて見た実験

過剰なパラメータ：丸暗記と汎化

[Zhang et al., 2016]など

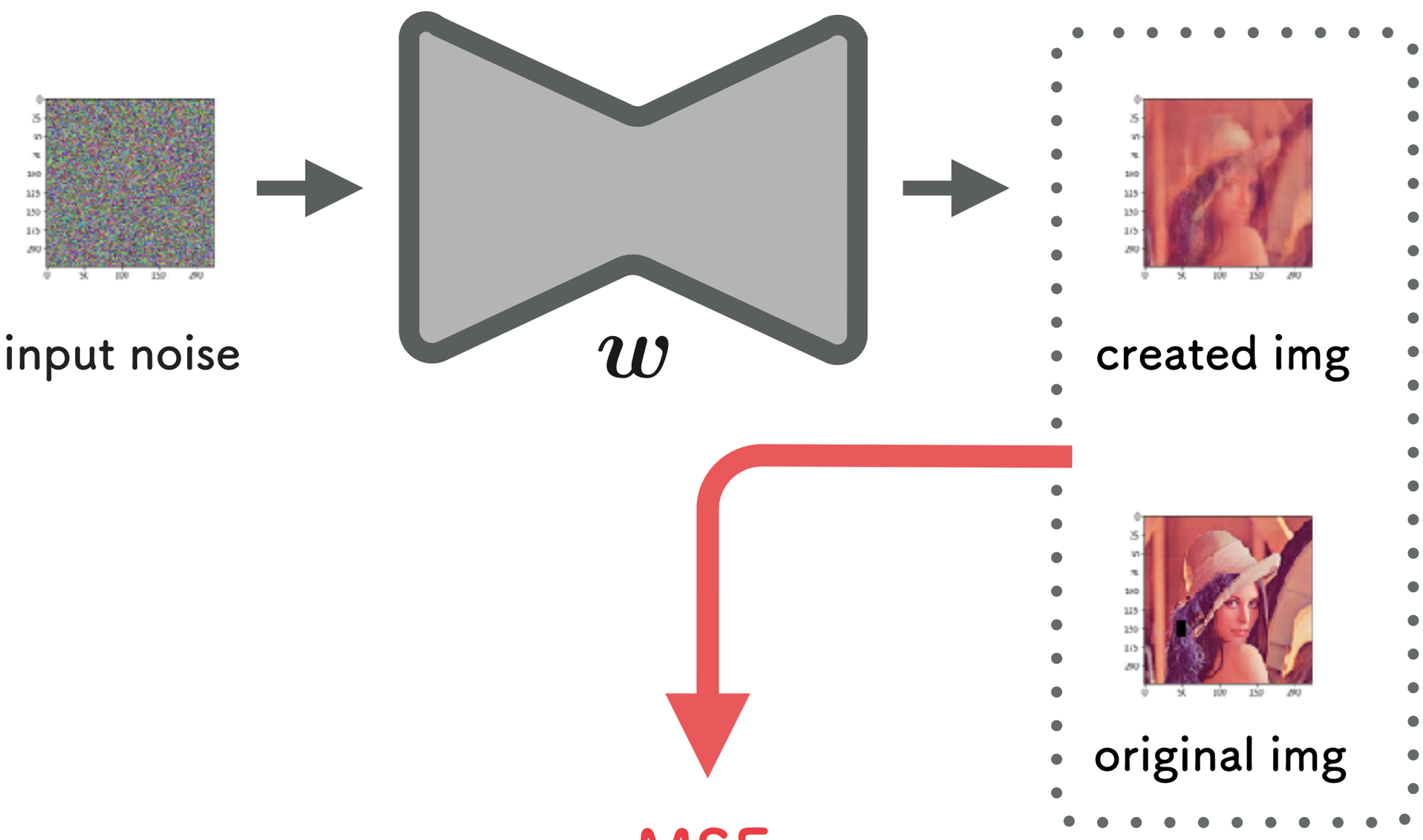
モデルの良さ(平均的な汎化能力)は、ネットワーク構造のデザインでほとんど決まっている。

不思議：丸暗記できるのに、パターンがあるときはしない！

認識困難な事例については暗記。これは無駄な学習になるので、正則化はこれ(single directions)を防ぐ。

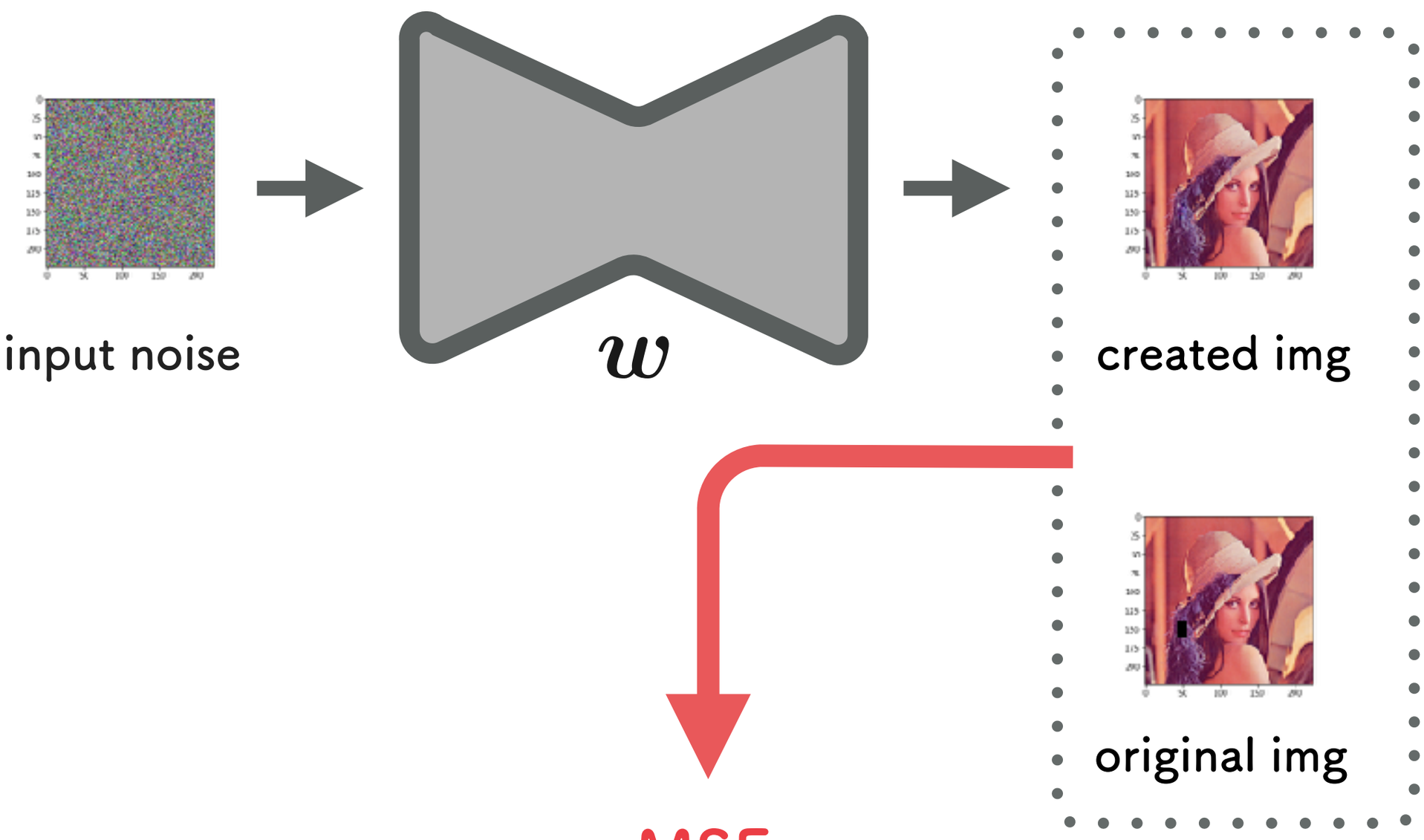
DNNにおけるモデルデザインの重要性を示唆する面白い結果：Deep Image Prior

Deep Image Prior [Ulyanov et al., 2017]



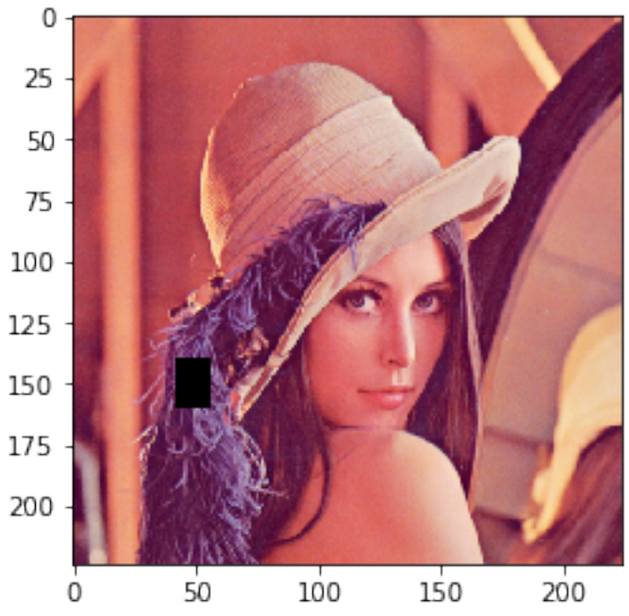
$$E(w) = \| (x(w) - x_0) \odot m \|^2$$

Deep Image Prior [Ulyanov et al., 2017]



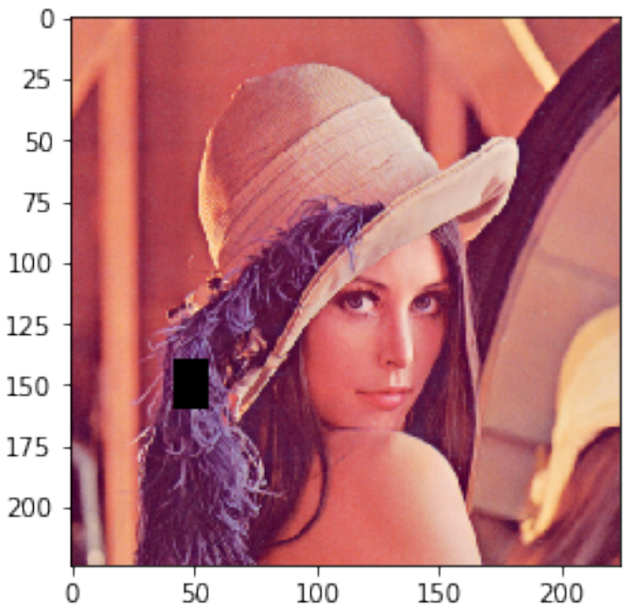
$$E(w) = \| (x(w) - x_0) \odot m \|^2$$

Deep Image Prior [Ulyanov et al., 2017]

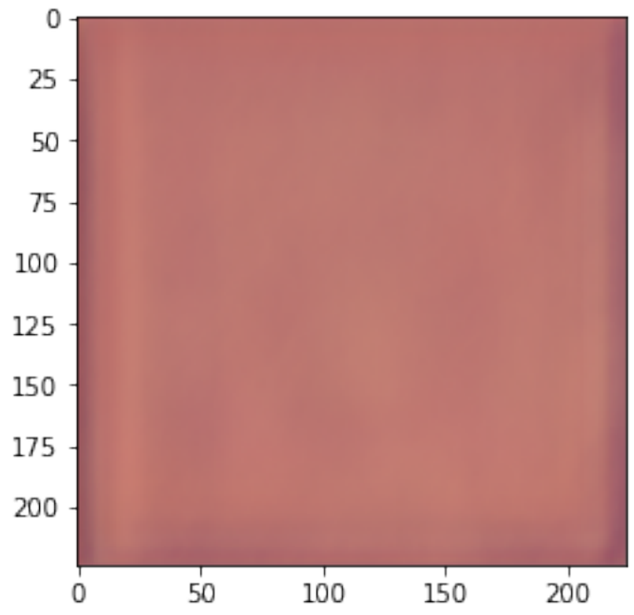


original

Deep Image Prior [Ulyanov et al., 2017]

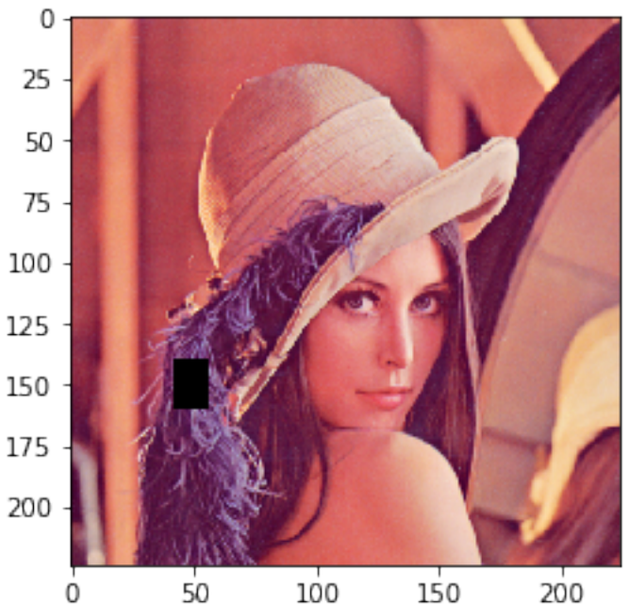


original

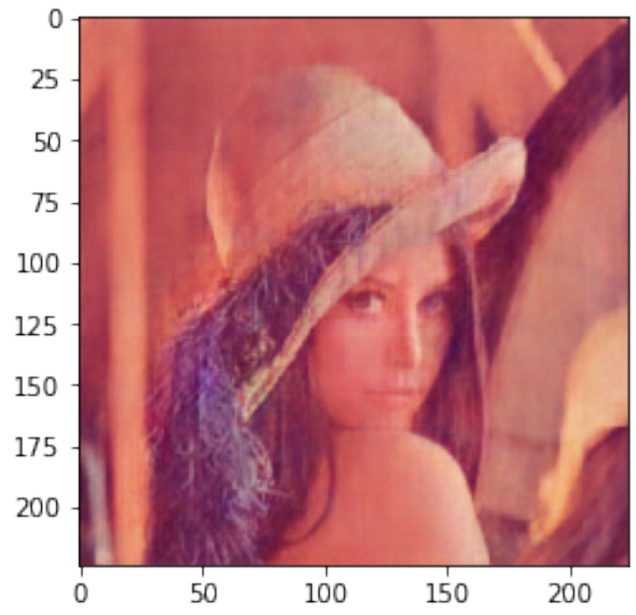


100 epochs

Deep Image Prior [Ulyanov et al., 2017]

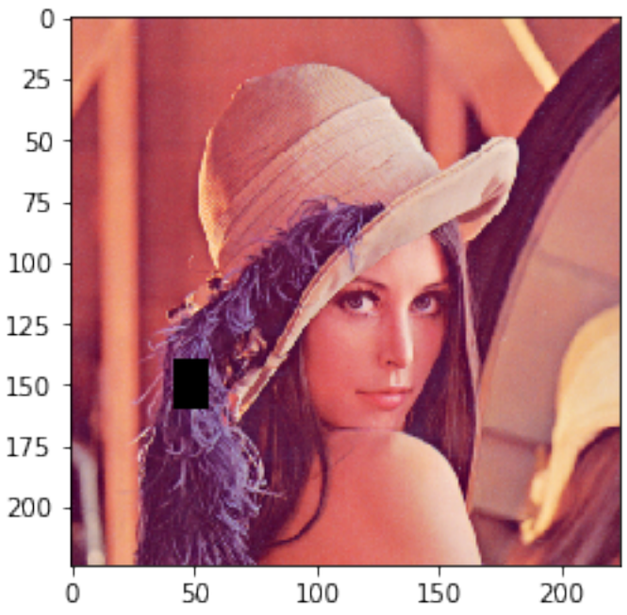


original

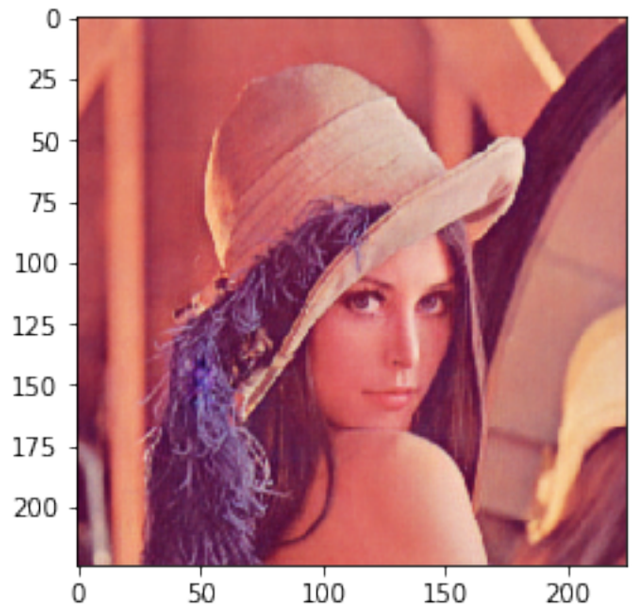


500 epochs

Deep Image Prior [Ulyanov et al., 2017]

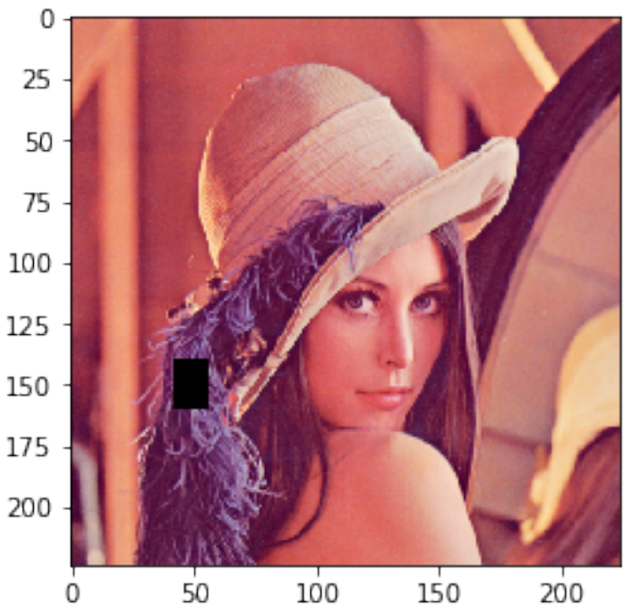


original

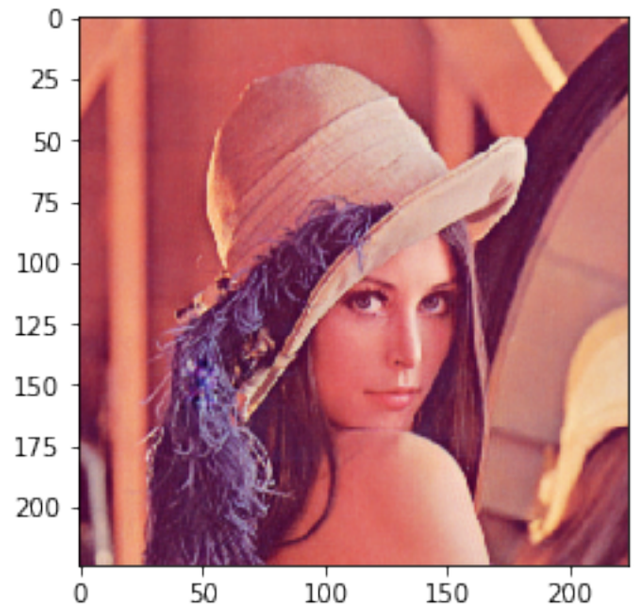


700 epochs

Deep Image Prior [Ulyanov et al., 2017]

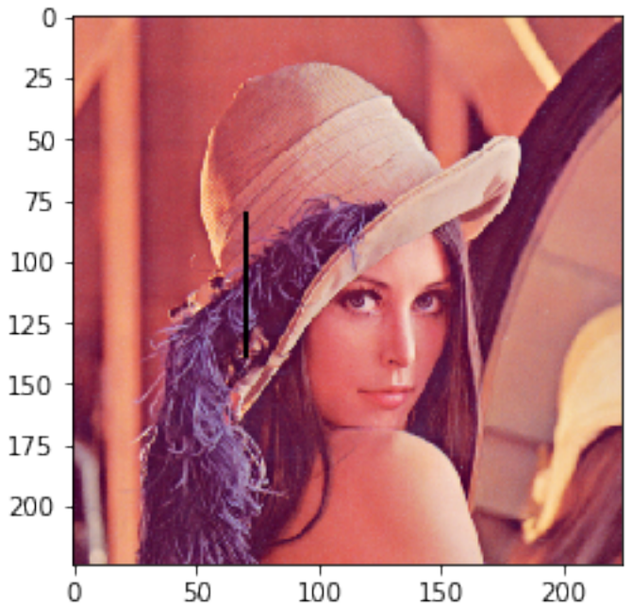


original



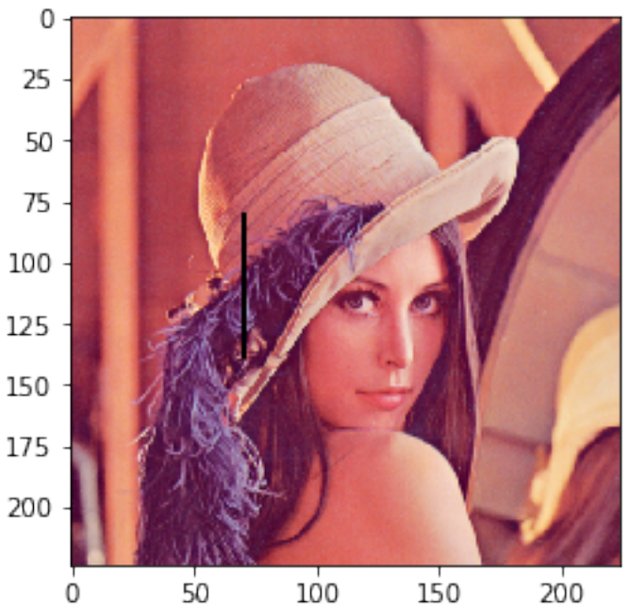
1000 epochs

Deep Image Prior [Ulyanov et al., 2017]

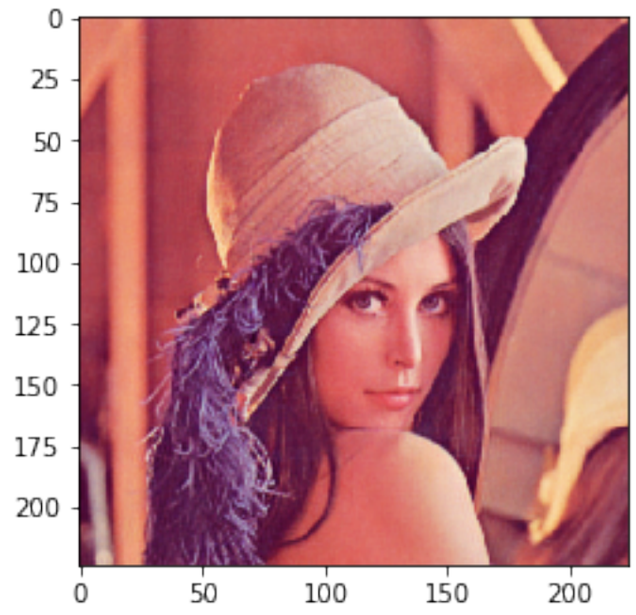


original

Deep Image Prior [Ulyanov et al., 2017]



original

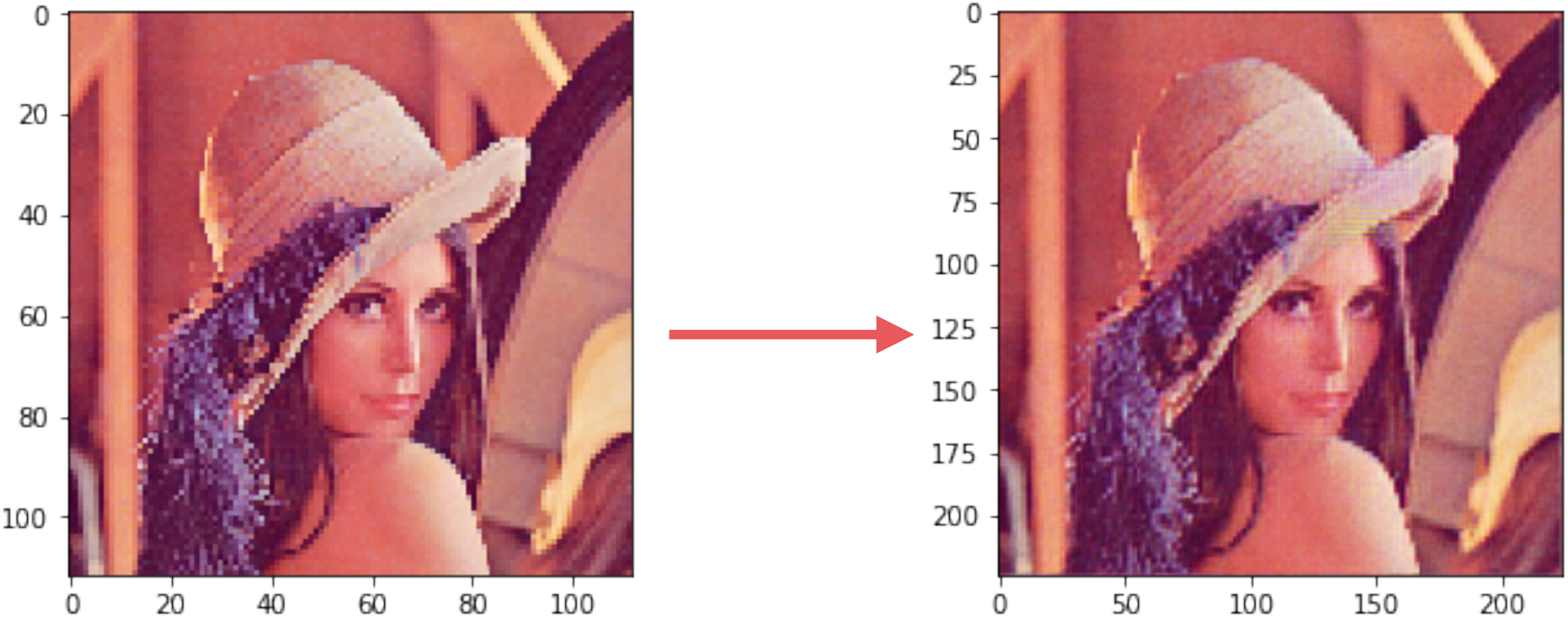


1000 epochs

Deep Image Prior [Ulyanov et al., 2017]

super-resolution（画質復元）もできる。

Deep Image Prior [Ulyanov et al., 2017]



Deep Image Prior [Ulyanov et al., 2017]

この一つのアイデアには数多くの応用があり、
かなり強力

Deep Image Prior [Ulyanov et al., 2017]

- 画像データは不要。ランダム初期値DNNから、一枚の画像だけで画質復元ができる

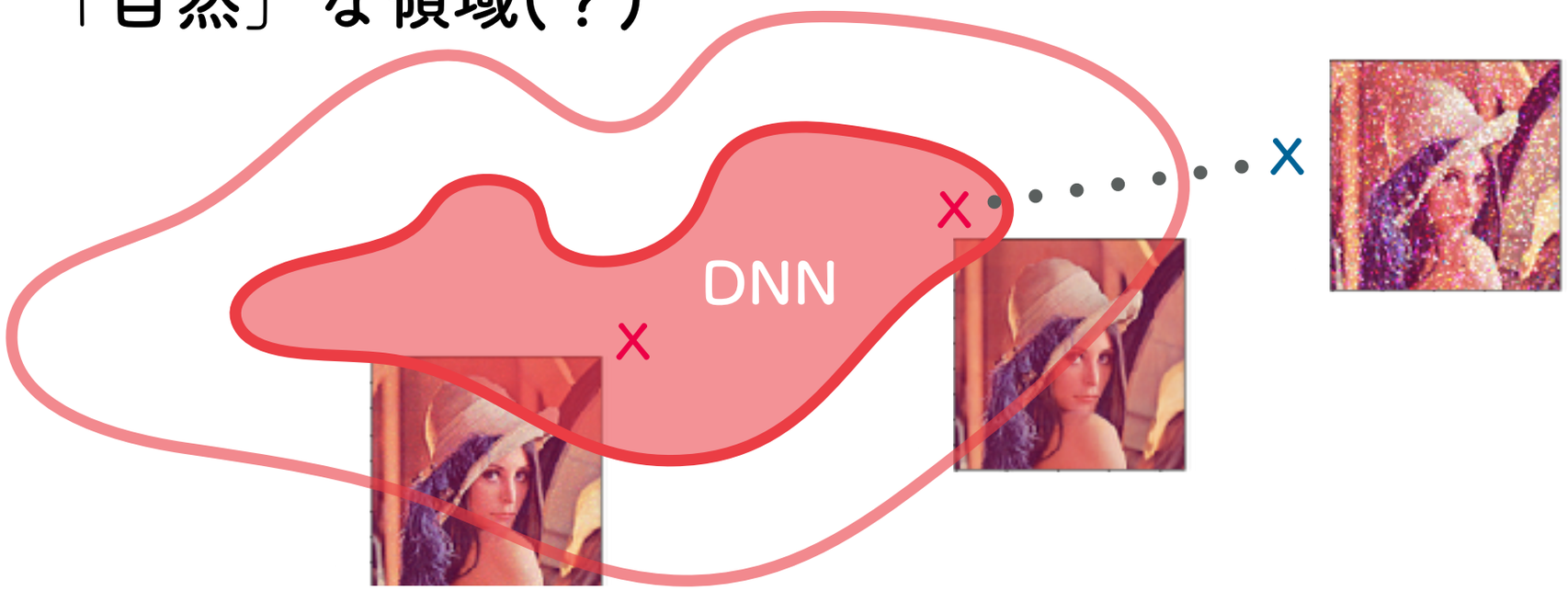
→ データセットに関するスケーラビリティではなく、アーキテクチャそのものの能力

Deep Image Prior [Ulyanov et al., 2017]

- 画像データは不要。ランダム初期値DNNから、一枚の画像だけで画質復元ができる

→ データセットに関するスケーラビリティではなく、アーキテクチャそのものの能力

- DNN approximatorがカバーする領域は、画像に関しては「自然」な領域(？)



続・汎化の謎

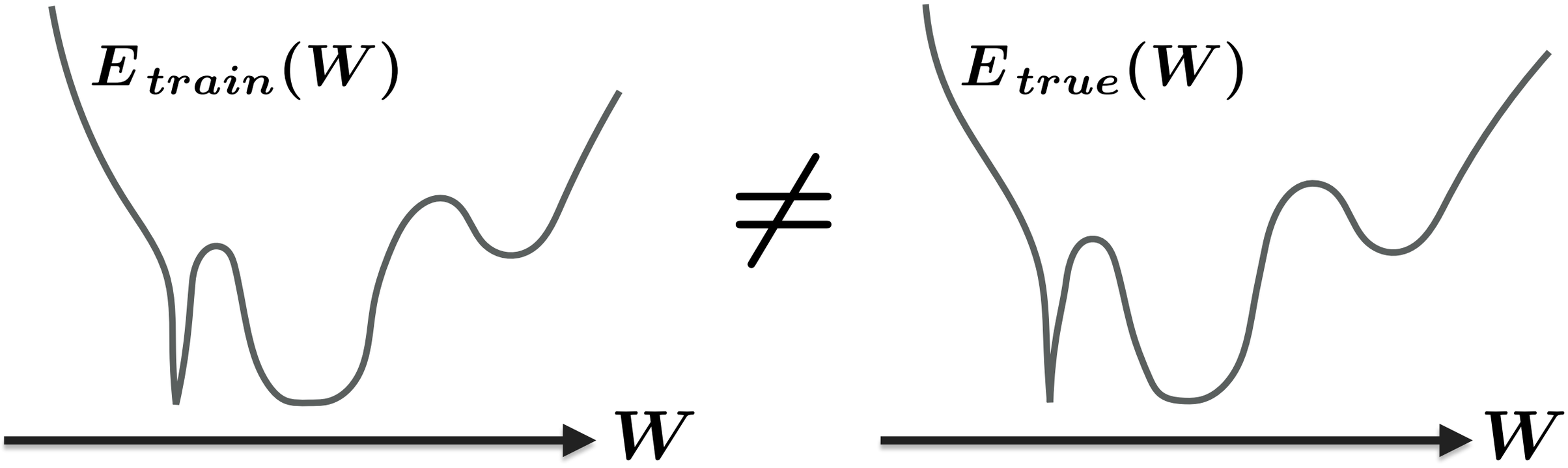
深層学習で得られる極小値には、良いものと悪いものがあるとして、その指標はなんでしょう？実は「平坦さ」というのが大事である可能性がわかってきました

最適化と平坦性

平坦性(flatness)・鋭度(sharpness)が汎化の基準？

手持ちの訓練用データ平均
で測った誤差関数

可能なすべてのデータ平均
で測った真の誤差関数

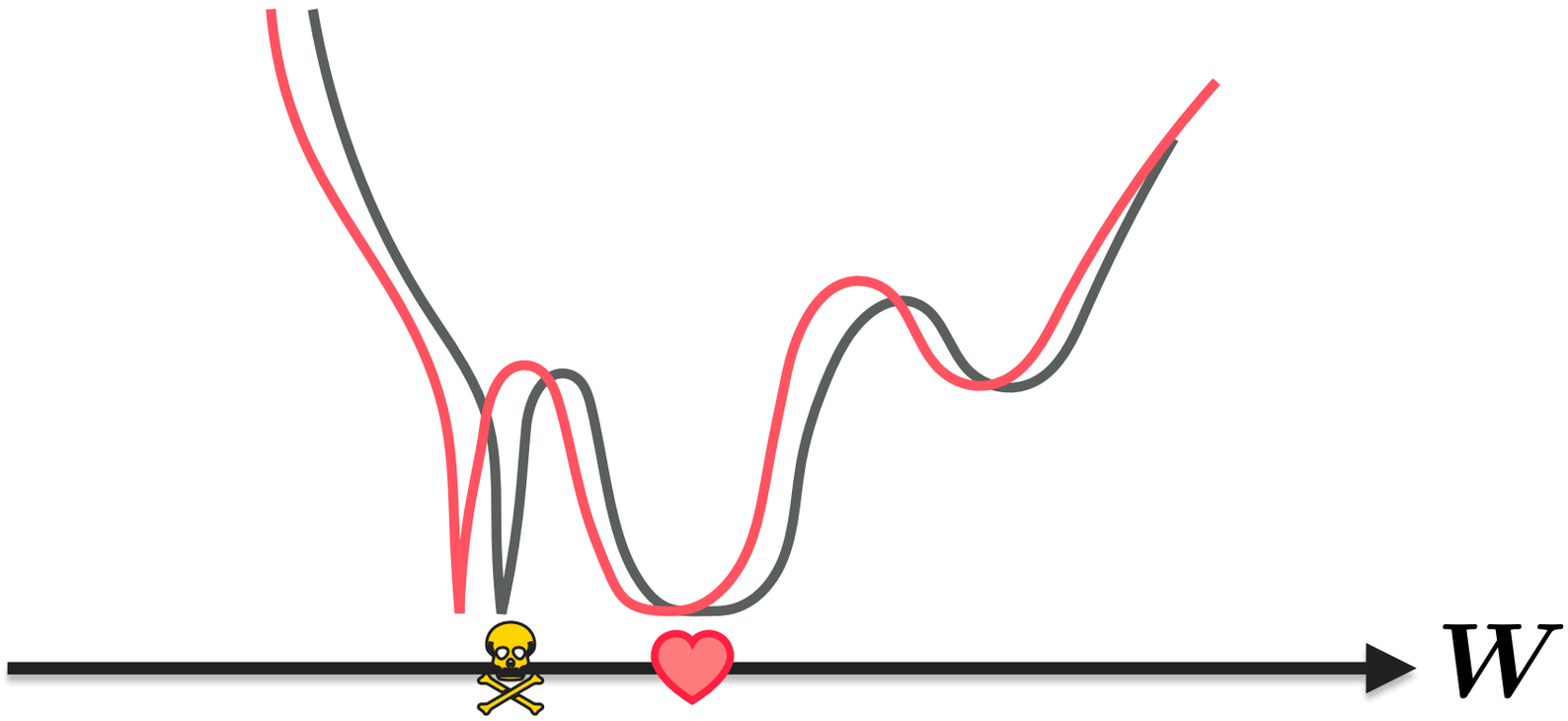


最適化と平坦性

平坦性(flatness)・鋭度(sharpness)が汎化の基準？

手持ちの訓練用データ平均
で測った誤差関数

可能なすべてのデータ平均
で測った真の誤差関数



SGD implies Flatness??

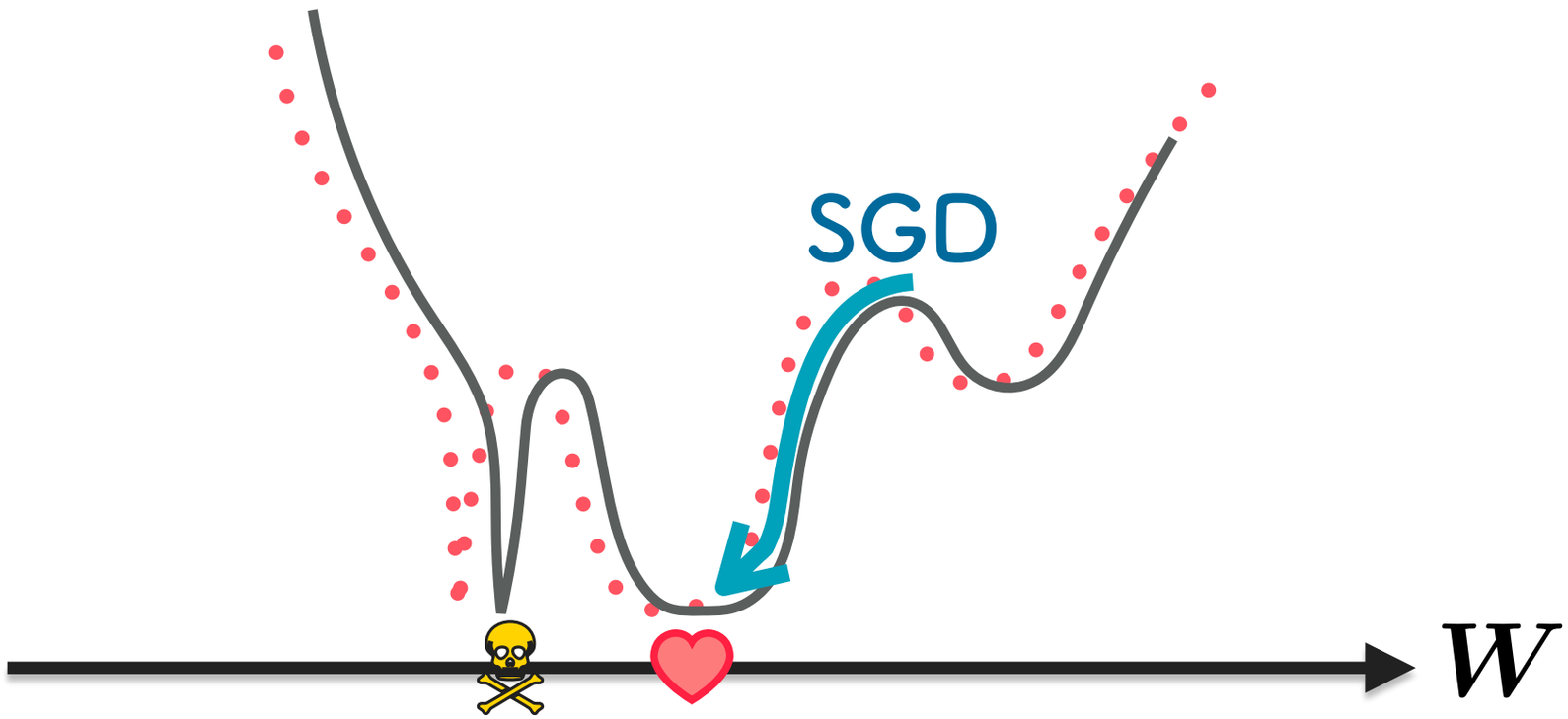
実は我々はこれまでそうとは気づかずに、平坦な
極小値を見つけやすい手法を使ってきた！？

SGD implies Flatness??

実は我々はこれまでそうとは気づかずに、平坦な
極小値を見つけやすい手法を使ってきた！？

手持ちの訓練用データ平均
で測った誤差関数

可能なすべてのデータ平均
で測った真の誤差関数



Flatness

しかし、平坦性は定義し難い！？

For any $\lambda > 0$

$$\max(0, \lambda x) = \lambda \max(0, x)$$

Flatness

しかし、平坦性は定義し難い！？

For any $\lambda > 0$

$$\max(0, \lambda x) = \lambda \max(0, x)$$

For ReLU activation $f(x) = \max(0, x)$

$$f(\lambda x) = \lambda f(x)$$

Flatness

Scale Invariance

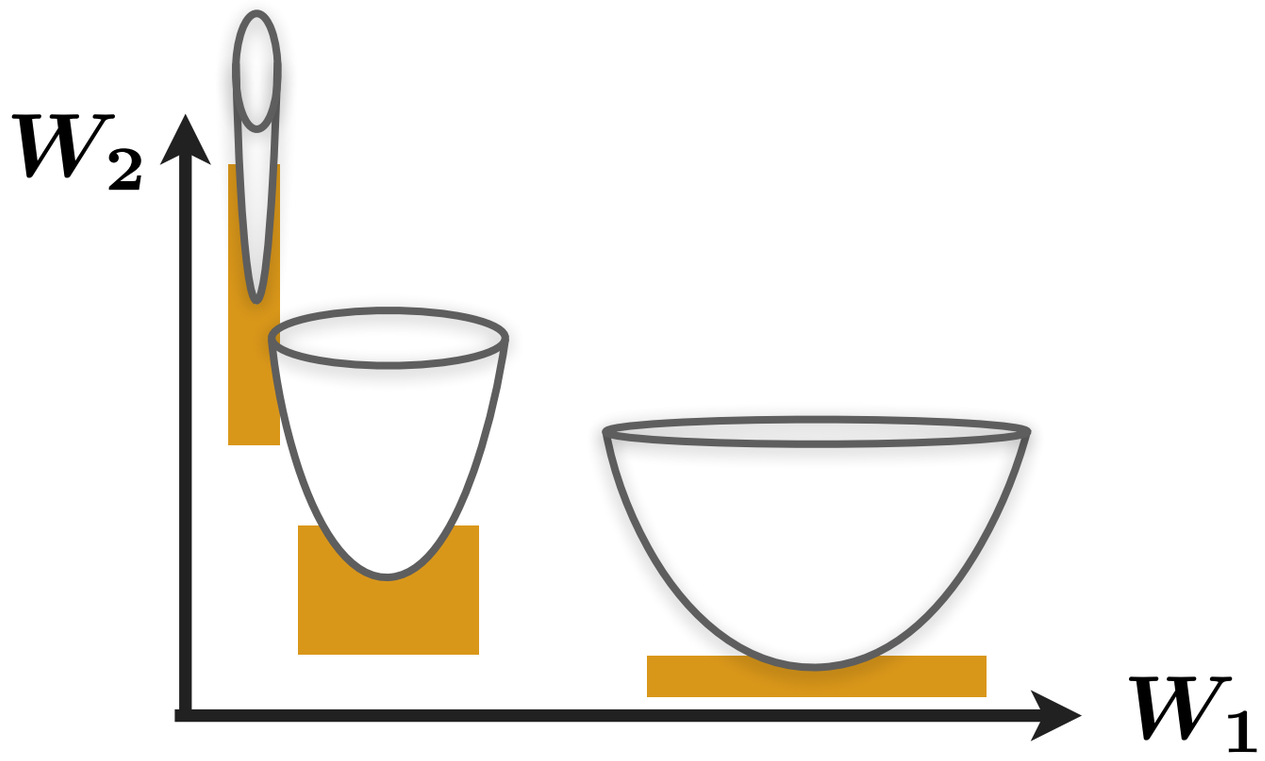
NN $$y = f\left(\boldsymbol{W_2} f(\boldsymbol{W_1} x)\right)$$

is invariant under scale transformation by $\lambda > 0$

$$\boldsymbol{W_1} \rightarrow \lambda \boldsymbol{W_1}$$

$$\boldsymbol{W_2} \rightarrow \lambda^{-1} \boldsymbol{W_2}$$

Flatness



Geometric flatness does not have meaning

$$W_1 \rightarrow \lambda W_1$$

$$W_2 \rightarrow \lambda^{-1} W_2$$

Find nice definition of ‘flatness’

汎化ギャップ

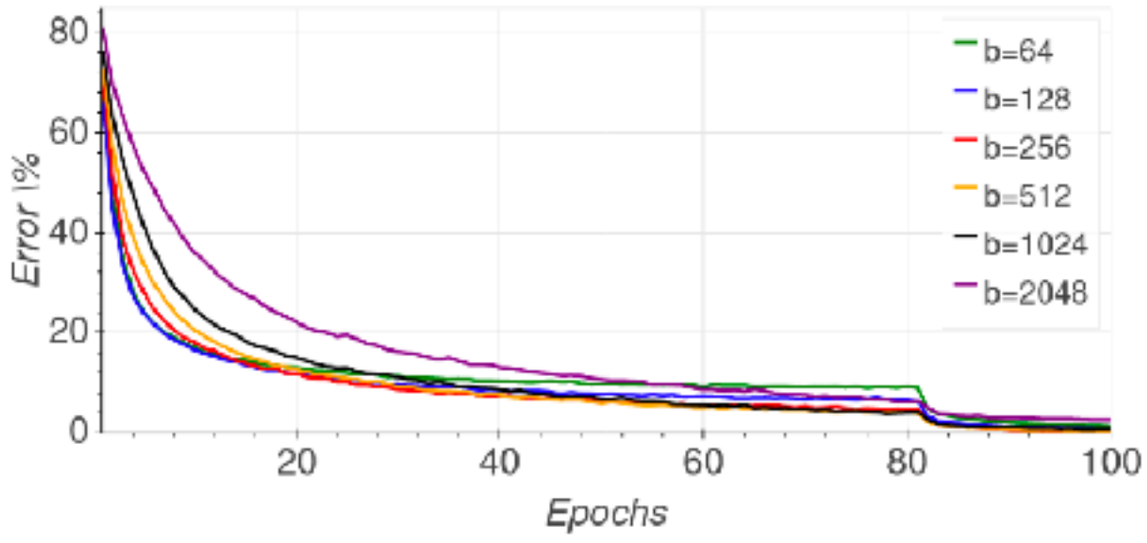
実は豊かな計算資源を使うと、出来上がったモデルの汎化性能下がってしまっているかもしれない、と言う問題

汎化ギャップ

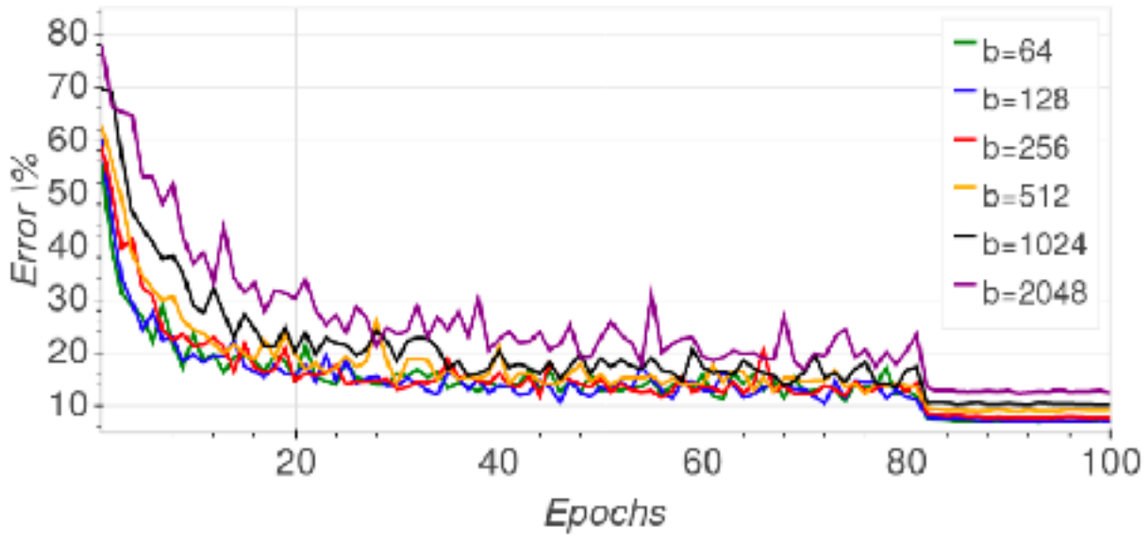
ミニバッチを小さくした方が収束が早い上に収束性能がよく、多量のGPUによる分散化が十分活用できない問題

Small mini-batch → converged model is generalized well

Large mini-batch → converged model is not generalized well



(a) Training error



(b) Validation error

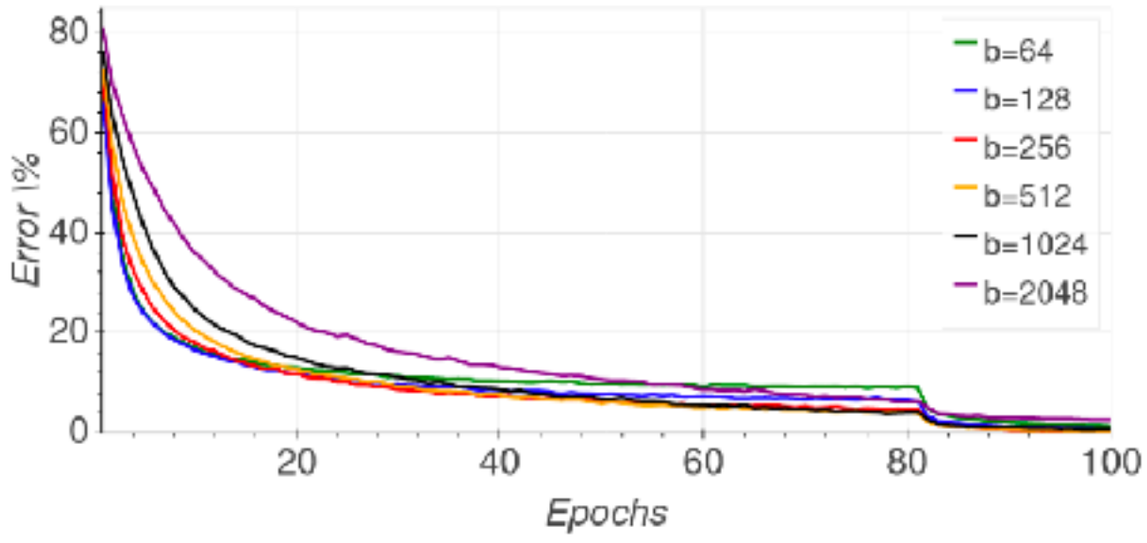
Figure1 of [Hoffer et al. '17]

汎化ギャップ

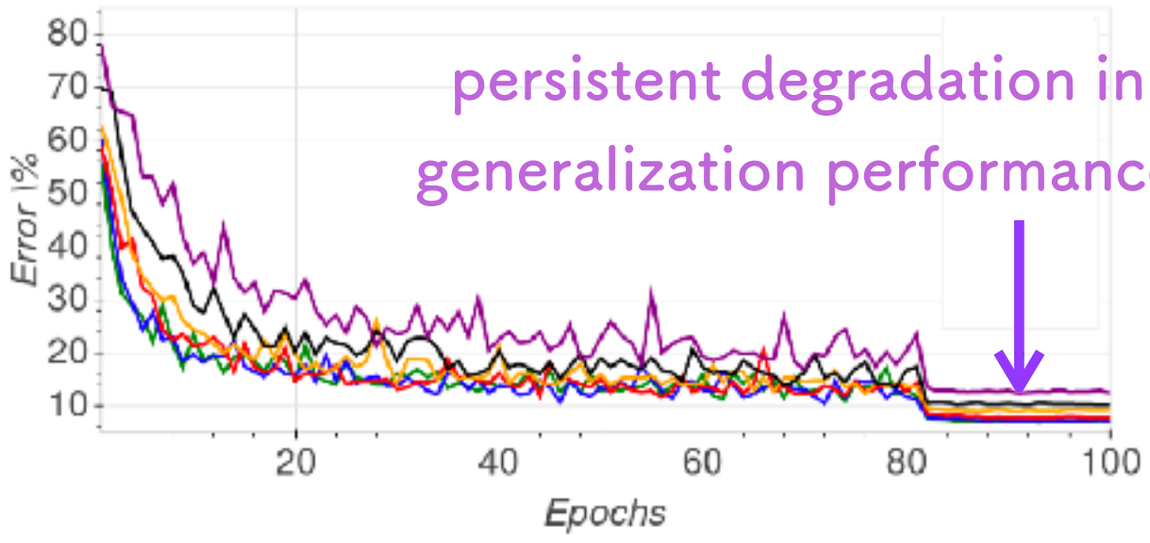
ミニバッチを小さくした方が収束が早い上に収束性能がよく、多量のGPUによる分散化が十分活用できない問題

Small mini-batch → converged model is generalized well

Large mini-batch → converged model is not generalized well



(a) Training error

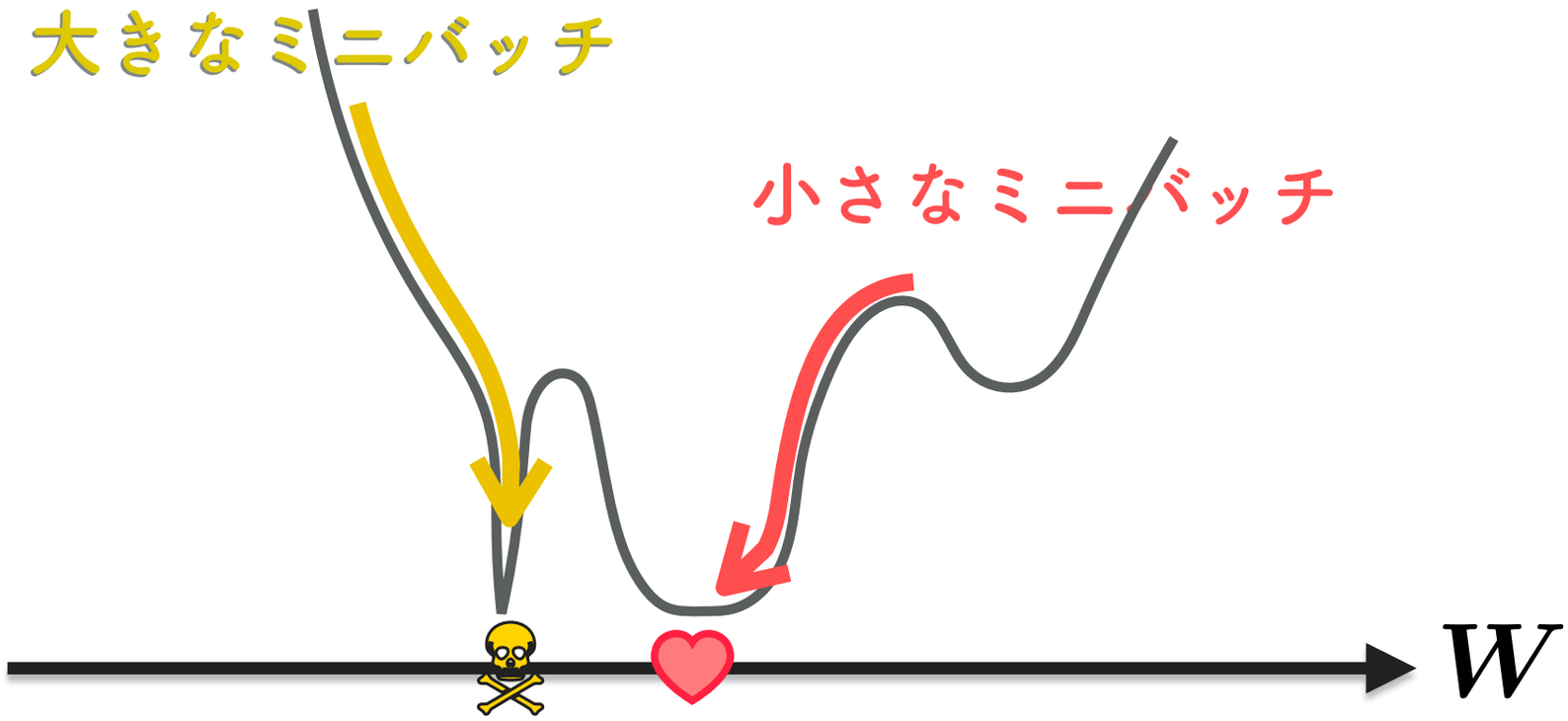


(b) Validation error

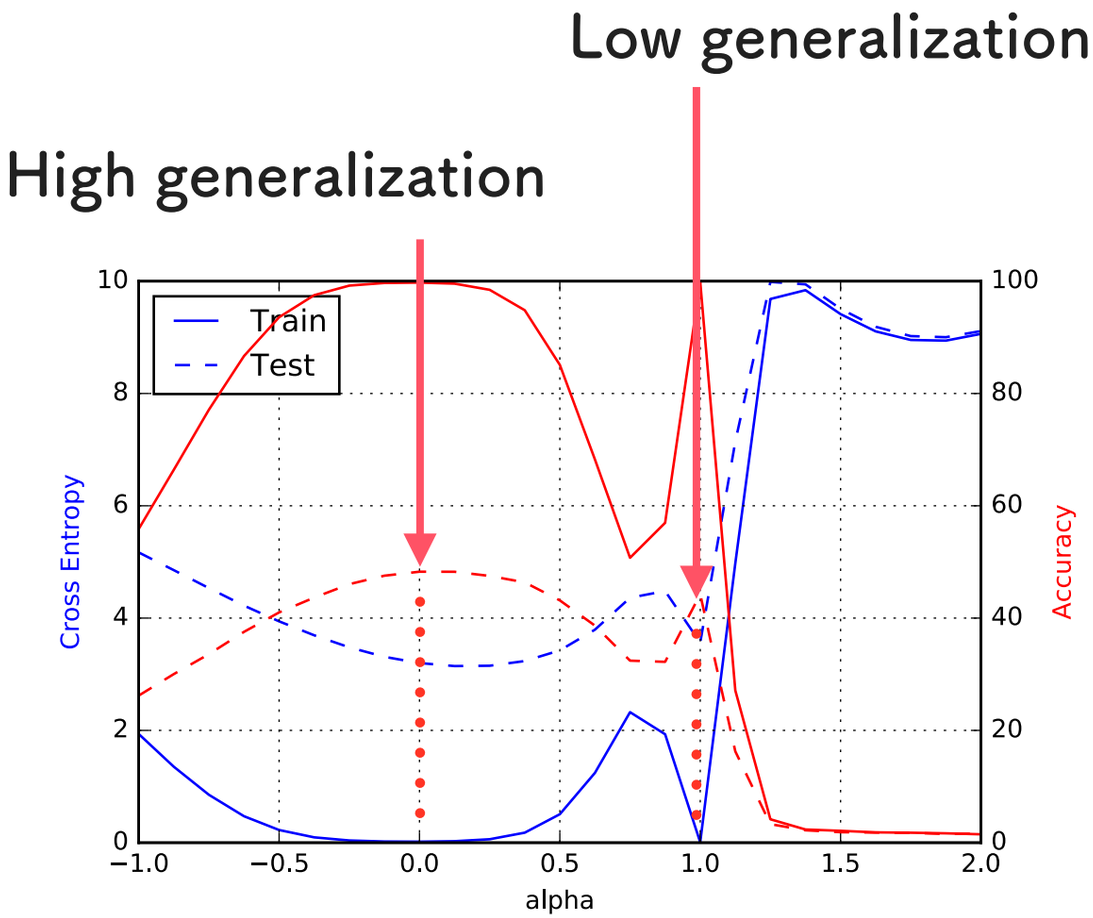
Figure1 of [Hoffer et al. '17]

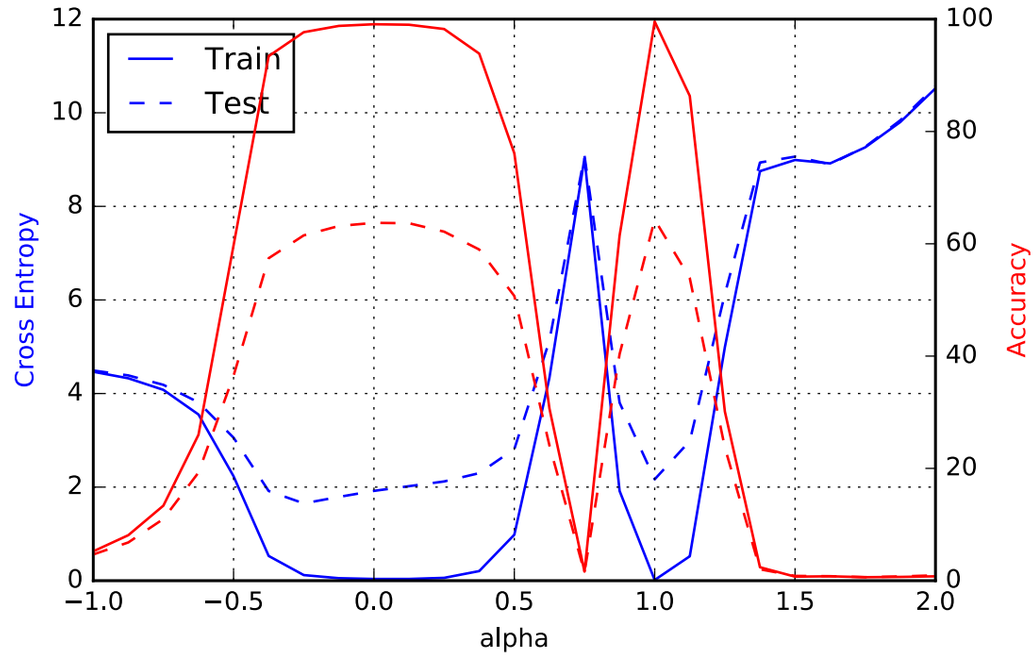
大きなバッチと汎化ギャップ

Hypothesis1: 鋭い極小値のために起こる



大きなバッチと汎化ギャップ





Flat　Sharp

empirical evidences

異常拡散と汎化ギャップ

汎化ギャップは、実はブラウン運動による拡散が異常に遅くなるプロセスと同じ。つまり単に収束に至っていないと言う可能性

異常拡散と汎化ギャップ

weight update via SGD $\boldsymbol{W}_{t+1} \leftarrow \boldsymbol{W}_t$

~ high-dim 'random walk on a random potential'
whose auto-covariance behaves : Hypothesis2

$$\mathbb{E}\left[\boldsymbol{E}(\boldsymbol{W}^{\mathbf{1}})\boldsymbol{E}(\boldsymbol{W}^{\mathbf{2}})\right] \sim \|\boldsymbol{W}^{\mathbf{1}} - \boldsymbol{W}^{\mathbf{2}}\|_{\mathbf{2}}^{\boldsymbol{\alpha}}$$

異常拡散と汎化ギャップ

weight update via SGD $\boldsymbol{W}_{t+1} \leftarrow \boldsymbol{W}_t$

~ high-dim 'random walk on a random potential'
whose auto-covariance behaves : Hypothesis2

$$\mathbb{E}\left[\boldsymbol{E}(\boldsymbol{W}^1)\boldsymbol{E}(\boldsymbol{W}^2)\right] \sim \|\boldsymbol{W}^1 - \boldsymbol{W}^2\|_2^{\alpha}$$

↓ [Bouchaud-Georges, Phys.Rep.]

$$\mathbb{E}\left[\|\boldsymbol{W}_t - \boldsymbol{W}_0\|_2^2\right] \sim (\log t)^{\frac{4}{\alpha}}$$

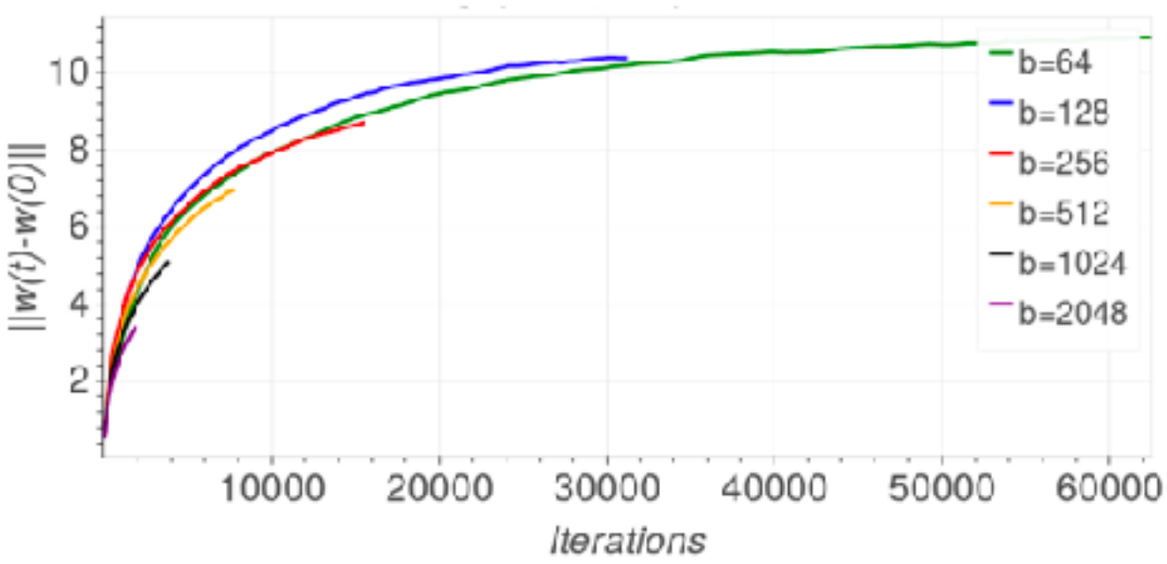
anomalously slow diffusion

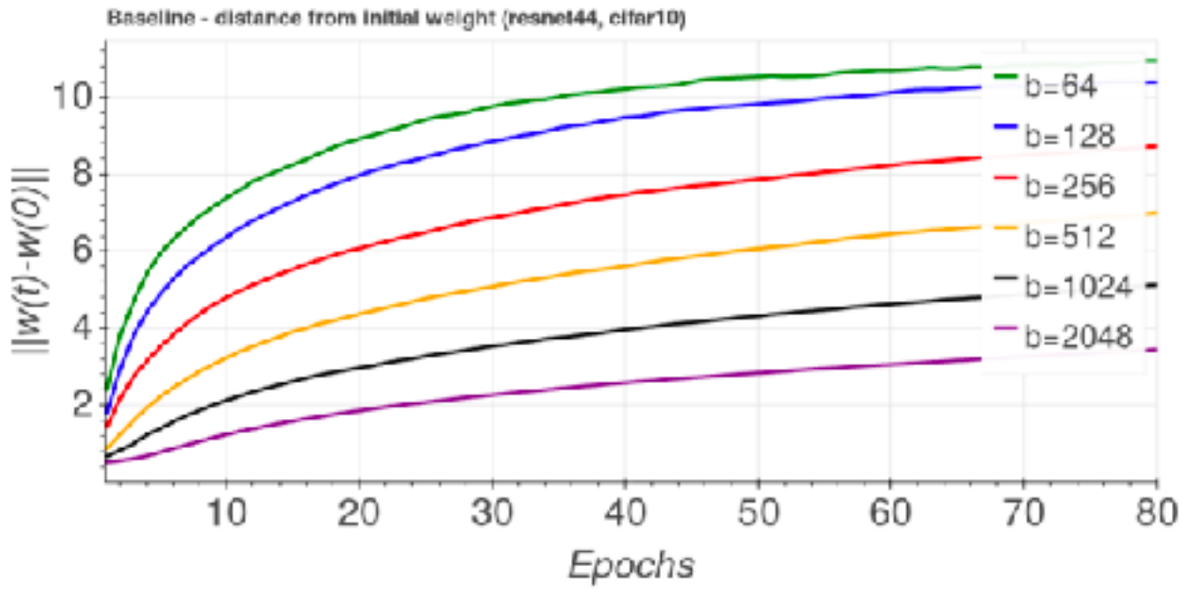
異常拡散と汎化ギャップ

distance from initial weight grows logarithmically with updates (not linear!!)

$$\|\boldsymbol{W}_t - \boldsymbol{W}_0\|_2 \sim \log t$$

* Deep Learning is $\alpha = 2$ experimentally.





match measure

[Hoffer et al. '17]

異常拡散と汎化ギャップ

distance from initial weight grows logarithmically with updates (not linear!!)

$$\|\boldsymbol{W}_t - \boldsymbol{W}_0\|_2 \sim \log t$$

* Deep Learning is $\alpha = 2$ experimentally.

Hypothesis2:

The gap comes from ‘random walk on a random landscape’ statistical model (ultra-slow diffusion)

汎化ギャップを閉じる

No inherent gap! More iterations!

Hypothesis2 → Ultra-Slow convergence

→ Continue iterations.

Generalization keeps improving (longer than thought) even without any observable improvement in training/validation errors!!

→ time-consuming & computationally costly …

別の理解

ヘッシアンのガウス・ニュートン近似

$$\boldsymbol{H}(w) \approx \boldsymbol{C}(w) + \bar{\boldsymbol{g}}(w)\bar{\boldsymbol{g}}(w)^{\top}$$

勾配の分散行列　　勾配の平均値のグラム行列

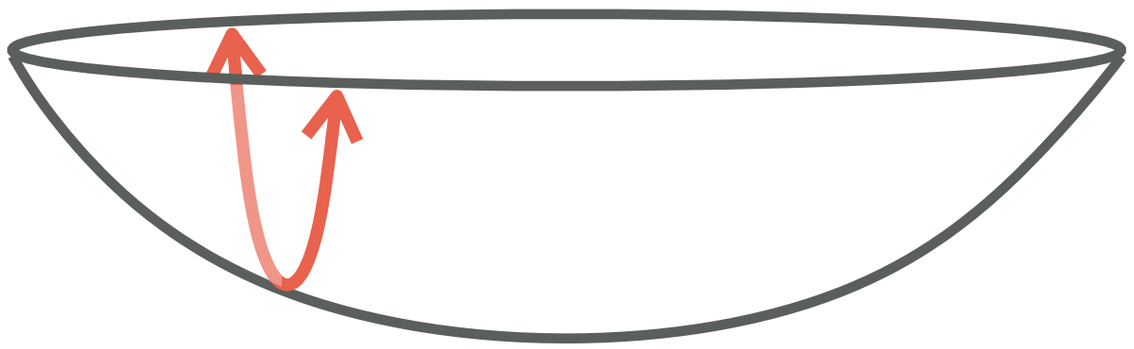
別の理解

ヘッシアンのガウス・ニュートン近似

$$\boldsymbol{H}(w) \approx C(w) + \bar{g}(w)\bar{g}(w)^\top$$

$$\approx \bar{g}(w)\bar{g}(w)^\top$$ 学習初期 [Scwartz et al. ‘17]

勾配がヘッシアンの最大固有値の方向を向いてしまう

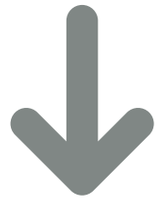


4. 汎化は平坦性か？

SGD as Orenstein-Uhlenbeck process

$$E = \frac{1}{N} \sum_{n} E_n$$

training error function



$$\hat{E} = \frac{1}{B} \sum_{n \in \mathcal{B}} E_n$$

minibatch estimate

SGD as Orenstein-Uhlenbeck process

$$\hat{g} = \nabla \hat{E}$$

estimate of gradient

$$w_{t+1} = w_t - \eta\, \hat{g}(w_t)$$

SGD as Orenstein-Uhlenbeck process

仮定：中心極限定理より

$$\hat{g}(w) \simeq g(w) + \frac{1}{\sqrt{B}}\Delta g$$

$$\Delta g \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$$

↓

$$\Delta w_t = -\eta\, g(w_t) + \sqrt{\frac{\eta}{B}}\sigma \Delta X$$

$$\Delta X \sim \mathcal{N}(0, \sigma \mathbb{I})$$

SGD as Orenstein-Uhlenbeck process

連続極限を取ると、次の確率微分方程式

$$dw(t) = -g(w)dt + \sqrt{\frac{\eta}{B}}\sigma dX(t)$$

（ランジュバン方程式）

これに従う変数の分布は、フォッカー・プランク方程式で記述される

SGD as Orenstein-Uhlenbeck process

連続極限を取ると、次の確率微分方程式

$$dw(t) = -g(w)dt + \sqrt{\frac{\eta}{B}}\sigma dX(t)$$

（ランジュバン方程式）

局所最小値周りにだけ注目する：

$$E = \frac{1}{2}w^\top Hw \rightarrow g(w) = Hw$$

SGD as Orenstein-Uhlenbeck process

平行分布は次で与えられる：

$$P_{eq}(w) = P_0\, e^{-\frac{1}{2} w^\top S^{-1} w}$$

$$\propto e^{-\frac{E(w)}{2T}}$$

$$SH + HS = \frac{\eta\sigma^2}{B}\mathbb{I} \rightarrow T = \frac{\eta\sigma^2}{B}$$

SGD as Orenstein-Uhlenbeck process

ある区間にパラメータがある確率は

$$\int_I dw\, P_{eq}(w) \simeq \frac{1}{\sqrt{\det H}}\, e^{-\frac{E(0)}{2T}}$$

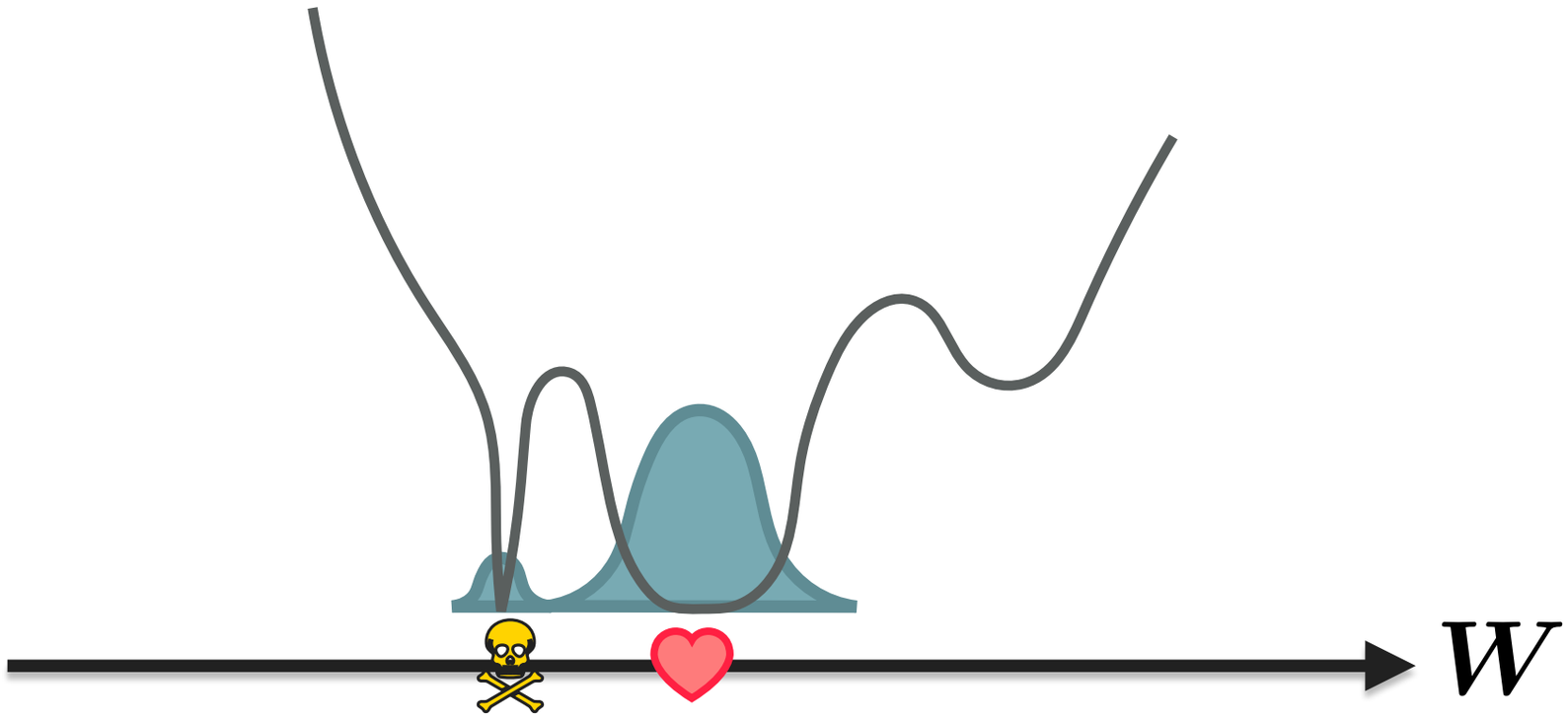
ラプラス原理(大偏差原理)を使った

$$T = \frac{\eta\sigma^2}{B}$$

低温(大バッチ)だとヘッシアン項が主要：鋭い極小値が好まれる

SGD as Orenstein-Uhlenbeck process

フォッカープランク方程式によると、平坦な方に落ちやすい



$$T = \frac{\eta\sigma^2}{B}$$

↓

$$\frac{\eta}{B} = const$$ とすれば、同じ平坦性

大きな学習率で、ランダム性を補う

（大きな学習率で安定な学習を行うには、規格化が大事）

汎化と周波数

Spectral Bias

weight clip $\boldsymbol{W \leftarrow clip(W, K)}$

↓

NN output becomes Lipschitz continuous

↓

Fourier comp of NN output is bounded

$$|\hat{y}(\vec{k})| \leq \mathcal{O}\left(\cdots \frac{(KW)^D}{\vec{k}^2}\right)$$

Spectral Bias

High frequency patterns are not learned

↓

Hard to memorize delta-function like special configuration

↓

Avoid memorization

5. まとめ

話がごちゃごちゃしてしまいましたが、

汎化は従来の統計的機械学習理論での標準的な議論では済まない？

モデルのデザインと丸暗記問題

平坦性か？そこが選ばれる理由は？

高次元のランドスケープ中の確率過程として正しく定式化したい