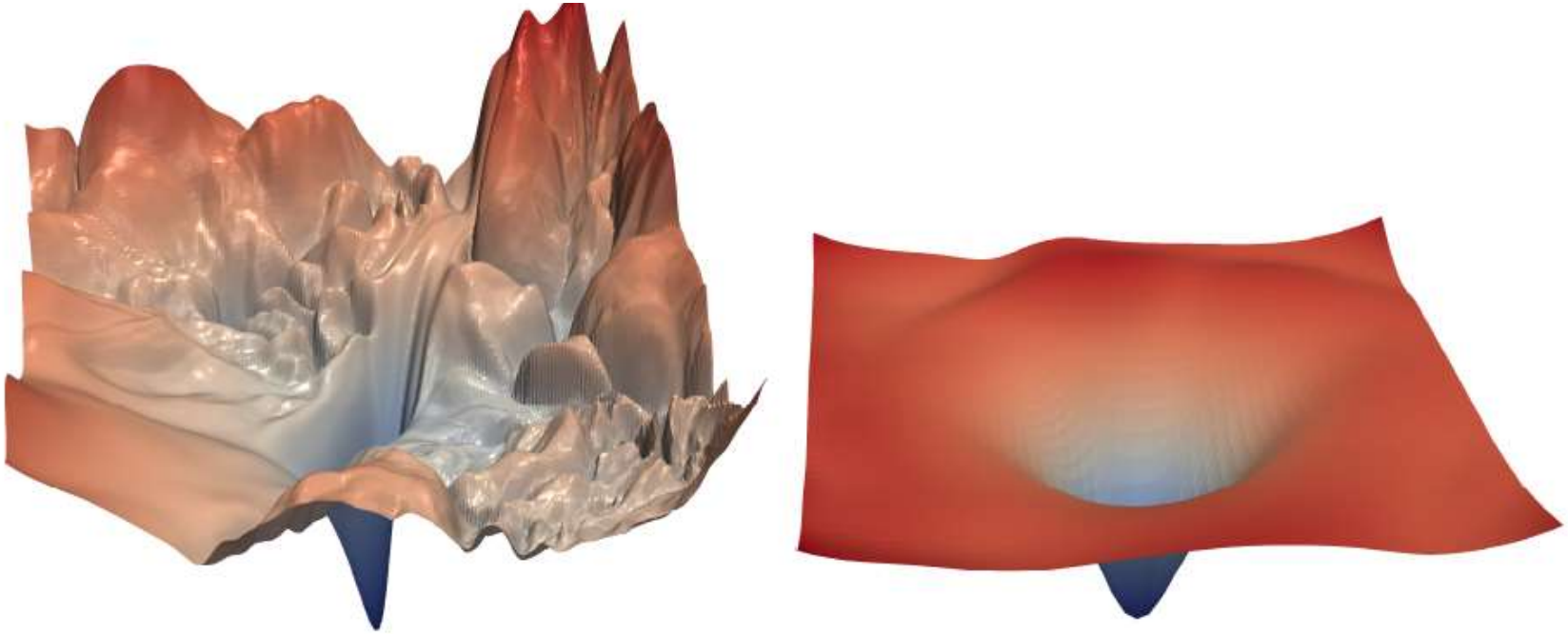
Visualizing the Loss Landscape of Neural Nets

Li, Hao, et al. “Visualizing the loss landscape of neural nets.” Advances in neural information processing systems 31 (2018). (NeurIPS2018)

この論文でやりたいこと

- 学習しやすいネットワーク、汎化性能の高いネットワークとは？
  アーキテクチャ・ハイパーパラメータ・他？
- Loss関数の形状を見ることで洞察を得られないか
- 例えば、Loss関数のFlatness/Sharpnessと、汎化性能に相関があるというこれまでの主張は本当か？

Loss関数ってどんなかたち？

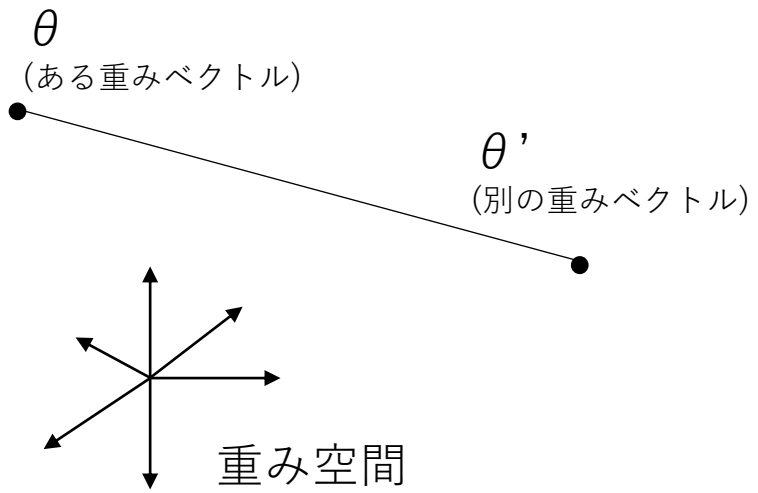


既存の取組み(1)

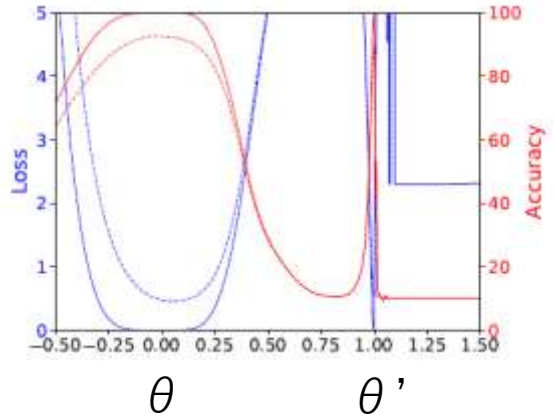
- 重み空間中の2点間を結ぶ直線上のLoss関数のプロファイルをとる [Goodfellowなど]

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m} \ell(x_i, y_i; \theta)$$

$$\theta(\alpha) = (1-\alpha)\theta + \alpha\theta'$$



プロット例



- 1次元では非凸性の評価は難しい
- スケールの不変性の影響を受ける(*後で理由を説明)

既存の取組み(2)

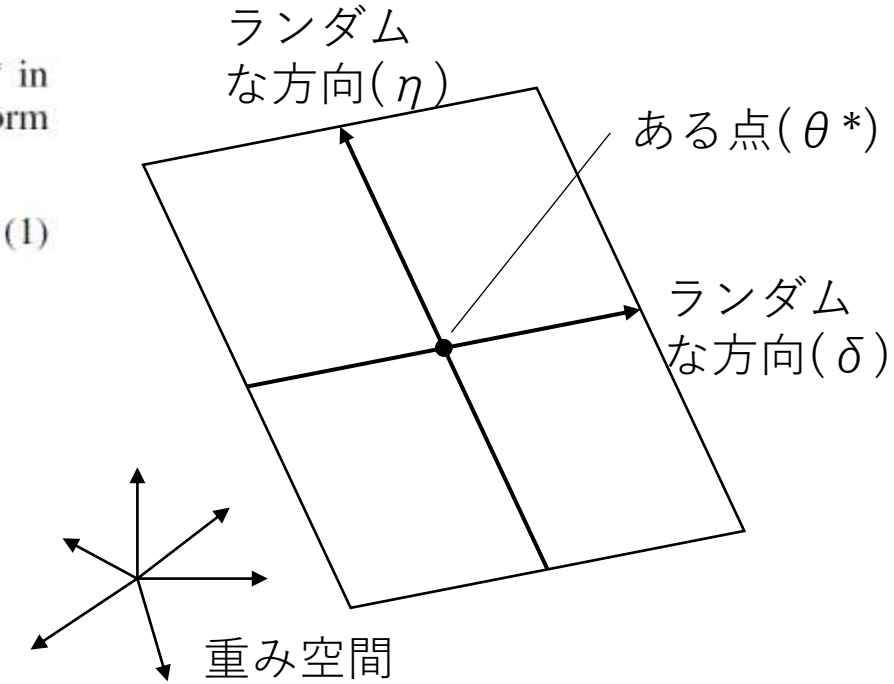
- ある点を中心に1～2次元のグリッドでLossプロファイルをとる

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m} \ell(x_i, y_i; \theta)$$

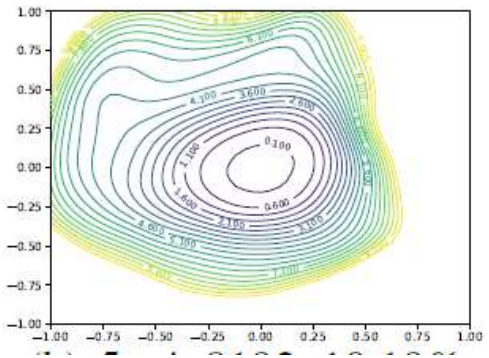
Contour Plots & Random Directions To use this approach, one chooses a center point θ^* in the graph, and chooses two direction vectors, δ and η. One then plots a function of the form $f(\alpha) = L(\theta^* + \alpha\delta)$ in the 1D (line) case, or

$$f(\alpha, \beta) = L(\theta^* + \alpha\delta + \beta\eta) \quad (1)$$

in the 2D (surface) case[2].



プロット例



- スケールの不変性の影響を受ける(*後で理由を説明)

既存手法の欠点

- スケール不変性のためLoss曲面を正しく抽出できない
  - 例)ReLUを考えた場合に、Layer間のScaleに不定性が発生する
    - $w_2 ReLU(w_1 x + b_1) + b_2 = w_2 w_1 x + w_2 b_1 + b_2$（層1の出力が正の場合）
    - w_2とw_1にスケールの不定性がある
  - 例)Batch normalizationがある場合にもScale不定性が発生する
    - BN前にスケールをかけても、正規化されてしまう

既存手法の欠点(続き)

- よく言われている仮説
  - 小さいバッチサイズのほうがLossの谷がフラットになり、よりフラットなほうが汎化性能が高い

- CIFAR-10を9層VGGで学習
  1. Batch size=128で学習
  2. Batch size=8192で学習

- 2つの最適化済み重み間の直線上のLossプロファイルをプロット

正則化をかけると、よりUpdate回数の多い小さいバッチサイズのほうがより重みが小さくなり、結果スケールの違いによりLossプロファイルのシャープ/フラットが逆転

(上段) Weight Decay(正則化)なし
(下段) Weight Decay(正則化)あり

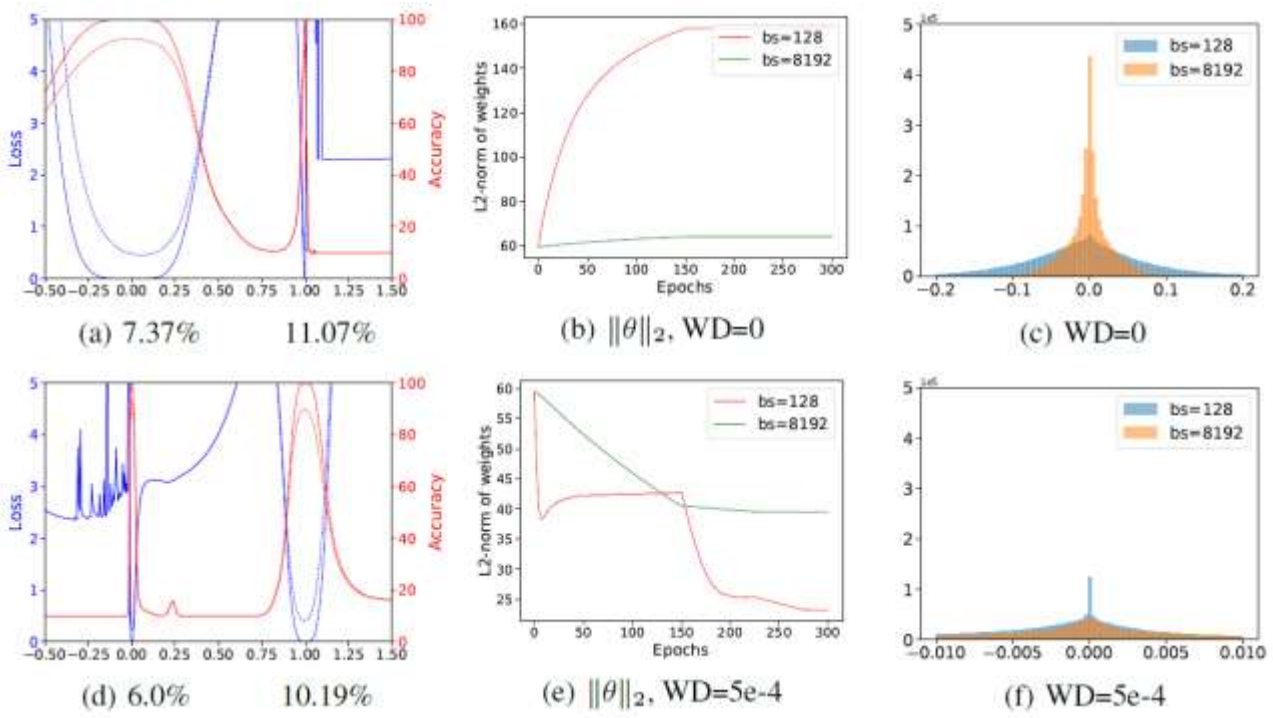


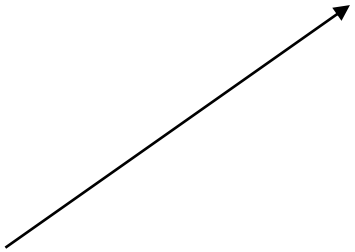
Figure 2: (a) and (d) are the 1D linear interpolation of VGG-9 solutions obtained by small-batch and large-batch training methods. The blue lines are loss values and the red lines are accuracies. The solid lines are training curves and the dashed lines are for testing. Small batch is at abscissa 0, and large batch is at abscissa 1. The corresponding test errors are shown below. (b) and (e) shows the change of weights norm $\|\theta\|_2$ during training. When weight decay is disabled, the weight norm grows steadily during training without constraints (c) and (f) are the weight histograms, which verify that small-batch methods produce more large weights with zero weight decay and more small weights with non-zero weight decay.

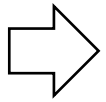
提案手法: Filter-wise normalization

- 選択したランダムな方向のうち、
  各層のフィルタ毎のノルムを中心点の重みベクトルのノルムと同じ
  大きさにする

we begin by producing a random Gaussian direction vector d with dimensions compatible with θ. Then, we normalize each filter in d to have the same norm of the corresponding filter in θ. In other words, we make the replacement $d_{i,j} \leftarrow \frac{d_{i,j}}{\|d_{i,j}\|}\|\theta_{i,j}\|$ where $d_{i,j}$ represents the jth filter (not the jth weight) of the ith layer of d, and $\|\cdot\|$ denotes the Frobenius norm. Note that the filter-wise normalization is different from that of [21], which normalize

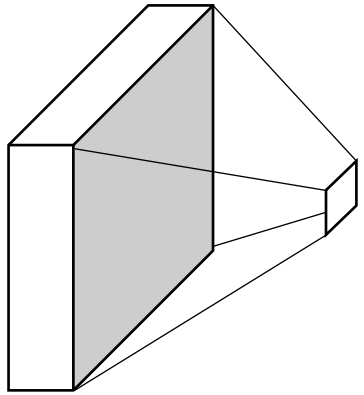
d: ランダムベクトル



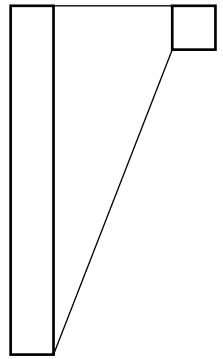


フィルタ毎の
部分空間毎に正規化

Convolution Filter



Fully Connected Filter



フィルタは
出力チャンネルもしくは出力次元
毎の重み係数

提案手法によるプロット

- 既存手法を用いた場合と同様の設定
  - CIFAR-10を9層VGGで学習
    1. Batch size=128で学習
    2. Batch size=8192で学習

- ただし、今回はランダムな方向を用いた手法で、それぞれの最適化周りのLossプロファイルをプロット

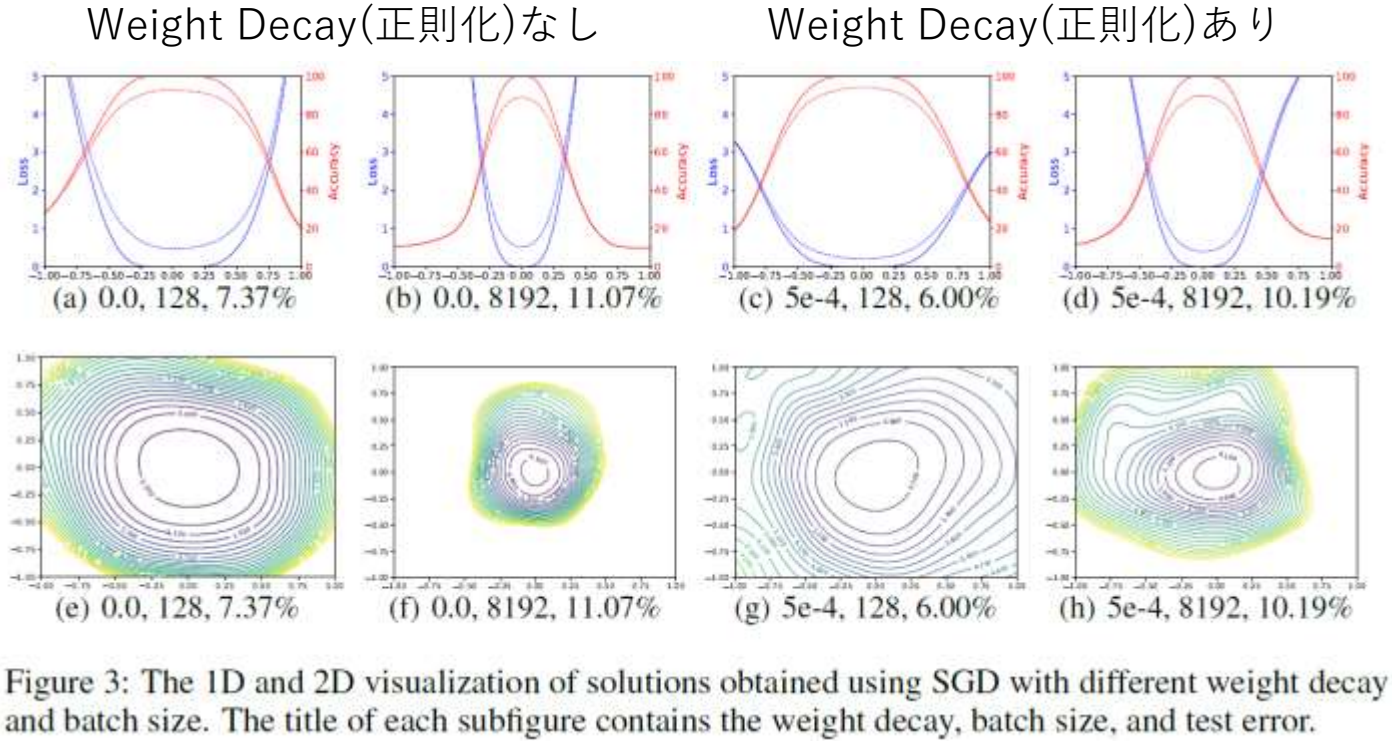


Figure 3: The 1D and 2D visualization of solutions obtained using SGD with different weight decay and batch size. The title of each subfigure contains the weight decay, batch size, and test error.

- 正則化を加えてもフラットネスとシャープネスの関係性が変わっていないことが分かる
- フラットな谷のほうがテストエラーが低く、汎化性能が高いと言える

ネットワークアーキテクチャの学習性や汎化性能への影響

- CIFAR-10を学習し、2次元のランダム平面上のLossプロファイルをプロット

- 比較したネットワークアーキテクチャ
  - ResNet: ResNet-20/56/110
  - ResNetからSkip connectを除去: ResNet-20/56/110-noshort
  - よりチャンネル数の多いResNet: Wide ResNet (x2, x4, x8 channels)

ネットワークの深さと
スキップコネクションの影響

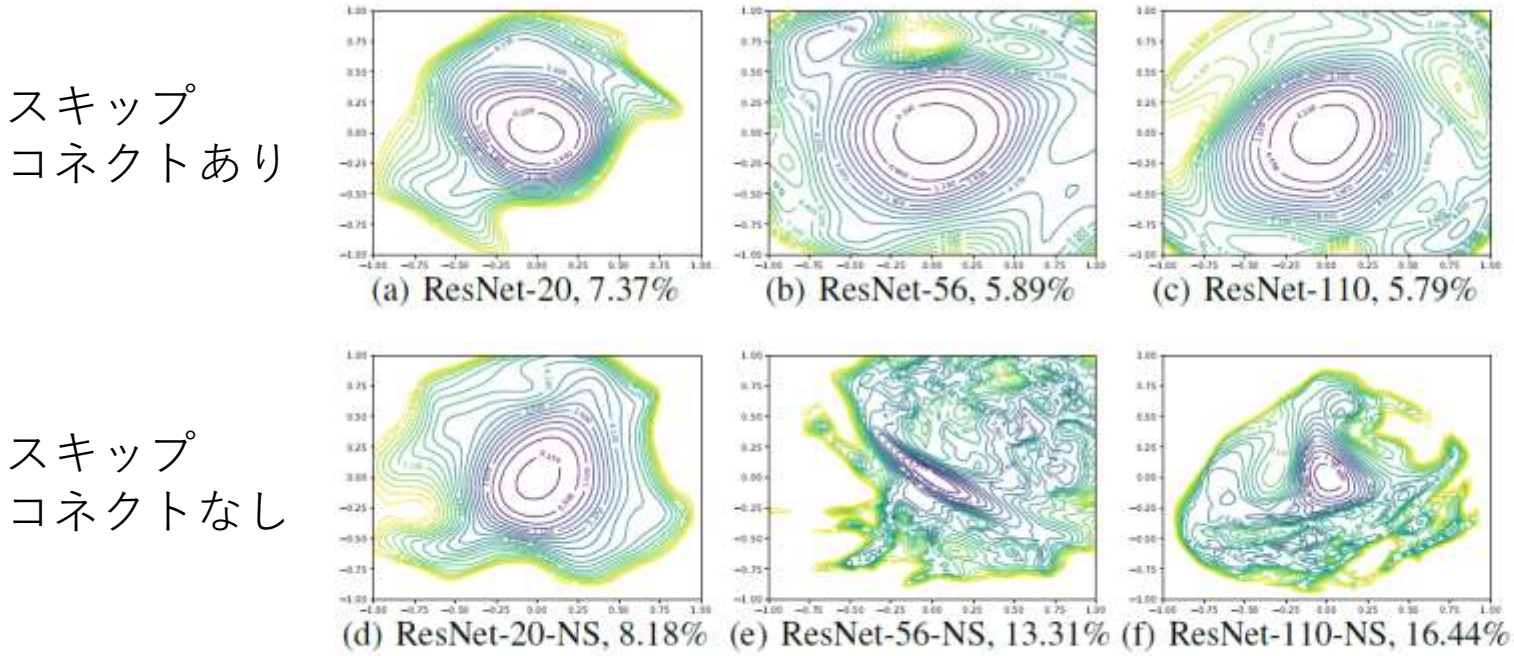


Figure 5: 2D visualization of the loss surface of ResNet and ResNet-noshort with different depth.

スキップコネクトがない場合、深くなるほど非凸性が高まり学習性が下がる
スキップコネクトによりその傾向が大きく緩和されていることが分かる

ネットワークの幅(ワイドさ)の影響

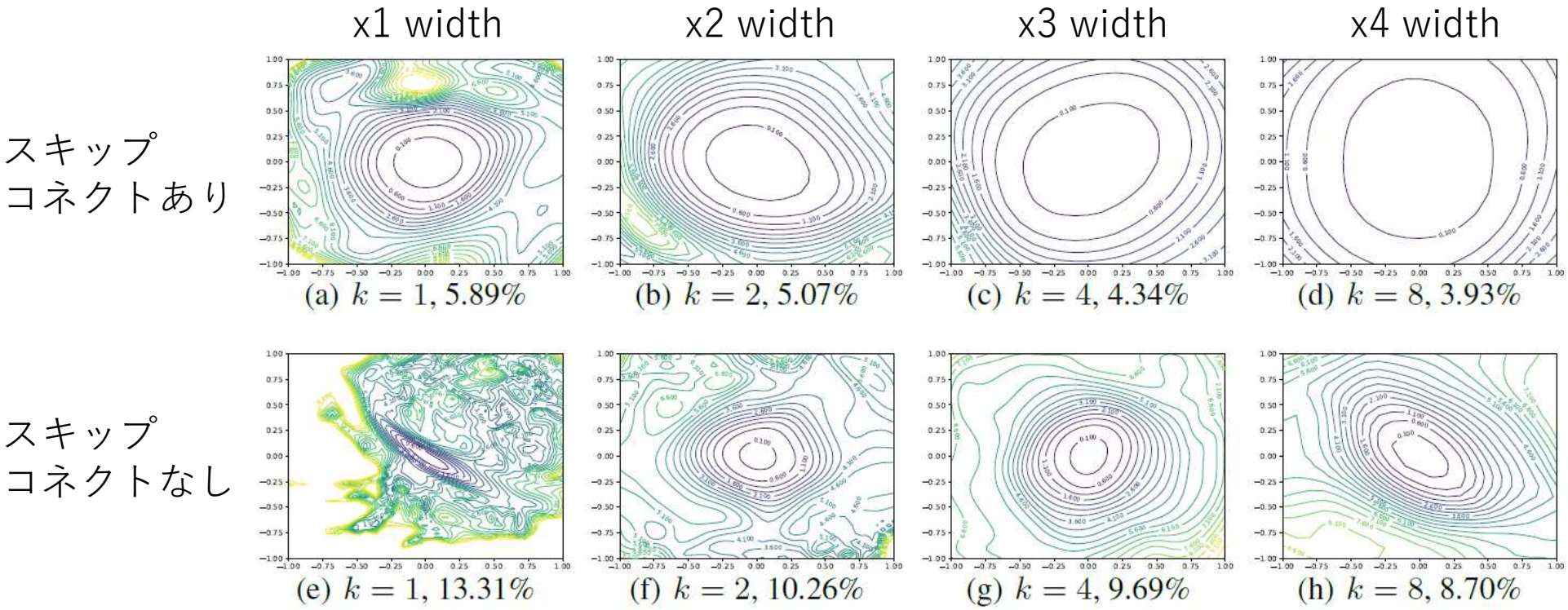


Figure 6: Wide-ResNet-56 on CIFAR-10 both with shortcut connections (top) and without (bottom). The label $k = 2$ means twice as many filters per layer. Test error is reported below each figure.

幅を広げることも、非凸性の緩和につながることが分かる
(*個人的な意見)プルーニングとの関連性も考えられそう

ネットワーク初期化

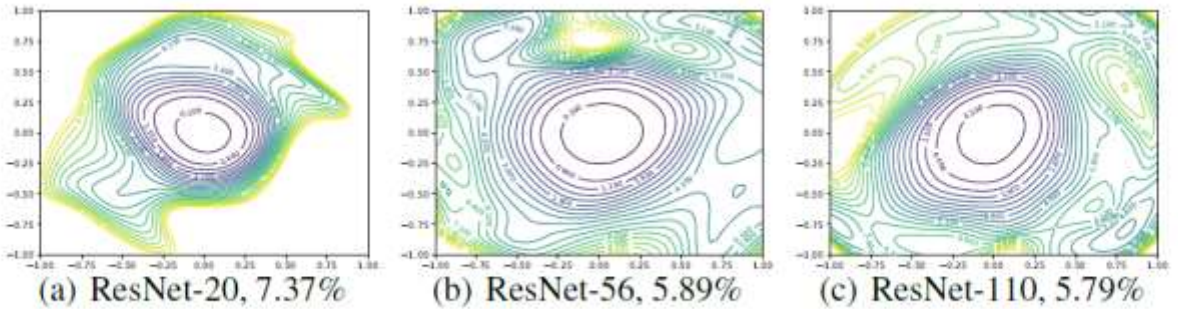
- スキップコネクトなしの場合のような、非凸性の高いネットワークでは初期値によっては、ローカルミニマムに捕まりやすい

- ⇒じゃあ、どうすればいいの？には答えてくれていない

汎化性能との関係

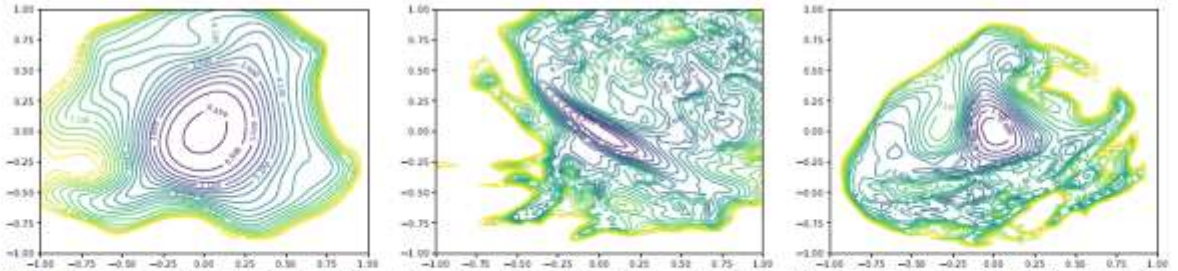
(再掲)

スキップ
コネクトあり



(a) ResNet-20, 7.37%　(b) ResNet-56, 5.89%　(c) ResNet-110, 5.79%

スキップ
コネクトなし



(d) ResNet-20-NS, 8.18%　(e) ResNet-56-NS, 13.31%　(f) ResNet-110-NS, 16.44%

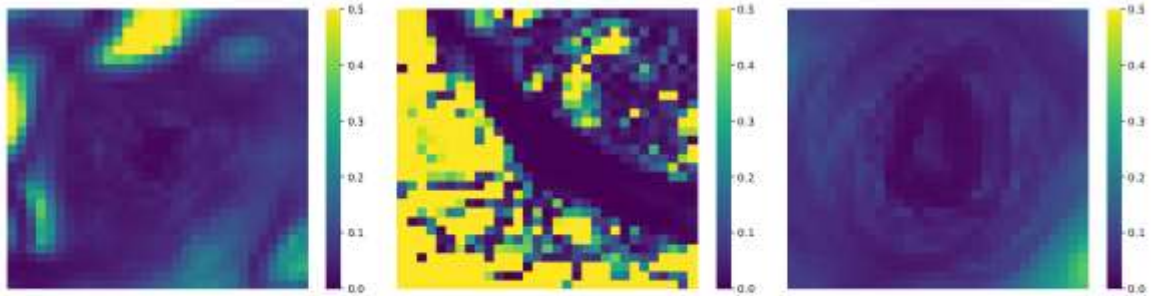
Figure 5: 2D visualization of the loss surface of ResNet and ResNet-noshort with different depth.

スキップコネクトなしの、カオスなLoss Landscapeだと、
テストエラーが大きくなり、汎化性も悪いことが分かる
(ほんと？⇒グローバルミニマムに辿り着けなかった可能性は？)

本当に凸／非凸性を見れているのか？

- 実際のLoss関数は超高次元であり、そのうちの高々2次元でしか見ていない
- ただ、ランダムな2次元プロットの曲率は、全次元での曲率の重み付き平均である、と示すことができる(導出は書いていない)
- Hessianの固有値
  - 凸／非凸性は、Hessian(2次微分)の固有値のSignで決まり、
  - 固有値の最小値と最大値は求めることができる(下記プロットは、最小値と最大値の比)
  - 最小値と最大値の比が小さいということは、負の固有値があったとしても小さい値だということ

that these visualizations fail to capture. To answer this question, we calculate the *minimum* and *maximum* eigenvalues of the Hessian, λ_{min} and λ_{max}.[4] Figure 7 maps the ratio $|\lambda_{min}/\lambda_{max}|$ across

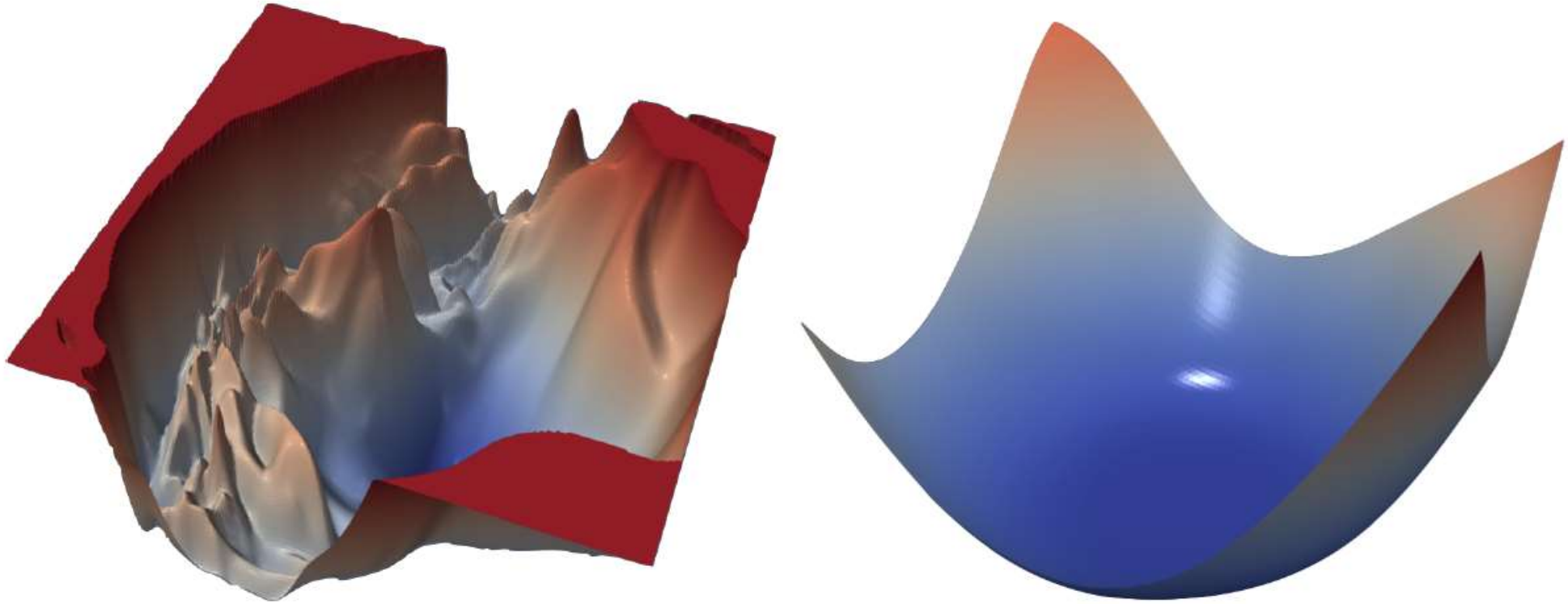


(a) Resnet-56 (b) Resnet-56-noshort (c) DenseNet-121

Figure 7: For each point in the filter-normalized surface plots, we calculate the maximum and minimum eigenvalue of the Hessian, and map the ratio of these two.

2次元ランダムプロットで凸性が高いプロットは、固有値比が低いことが分かる

3次元平面として見ると、こんな感じ



(a) ResNet-110, no skip connections

(b) DenseNet, 121 layers

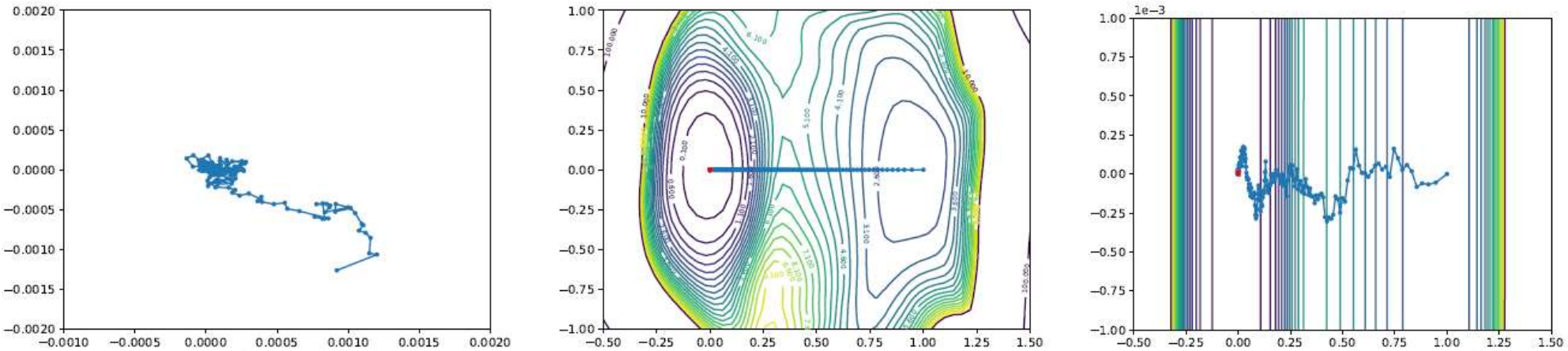
Figure 4: The loss surfaces of ResNet-110-noshort and DenseNet for CIFAR-10.

最適化パスのビジュアライズ

1. 最適化パスの、ランダム方向2次元プロットへの射影では、ほとんど意味のある動きを観測できない
2. また、初期値と最適値を結ぶ方向を1次元にとり、もう1次元をランダムにとる方法も、ランダム方向にはほとんど動いていない

1.の方法

2.の方法



(a) Two random directions (b) Random direction for y-axis (c) Enlarged version (b)

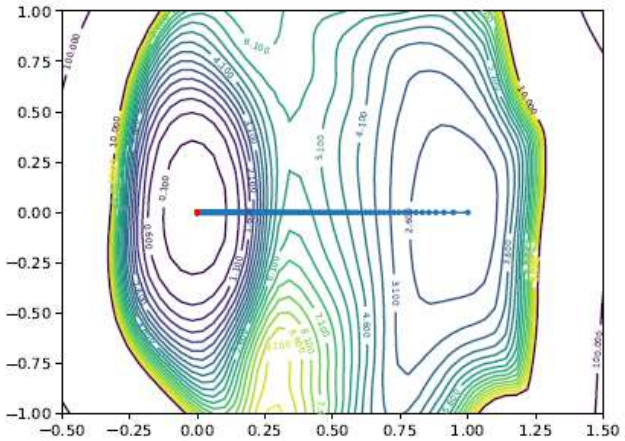
Figure 8: Ineffective visualizations of optimizer trajectories. These visualizations suffer from the orthogonality of random directions in high dimensions.

なぜうまくいかないか？

- 高次元になればなるほど、2つのランダムな方向は直交に近くなる

> It is well-known that two random vectors in a high dimensional space will be nearly orthogonal with high probability. In fact, the expected cosine similarity between Gaussian random vectors in n dimensions is roughly $\sqrt{2/(\pi n)}$ ([12], Lemma 5).

- もし、軌跡が非常に低い次元の部分空間内に存在している場合、殆どの方向はその部分空間に直交してしまい、動きをとらえることができない

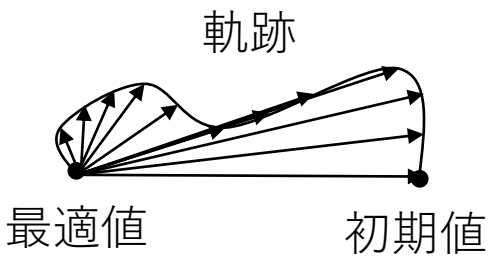


(b) Random direction for y-axis

この結果はそれを裏付けている

PCAを用いて部分空間を求める

- 最適化軌跡が張るベクトル群に、PCAをかけてもっともバリエーションの大きい方向を求める



途中点のベクトルが最も分散する方向をPCAで求める

- 小バッチサイズでは、SGDの影響でGradientに逆らって動いている
- 正則化も同様にGradientに逆らう方向に動く

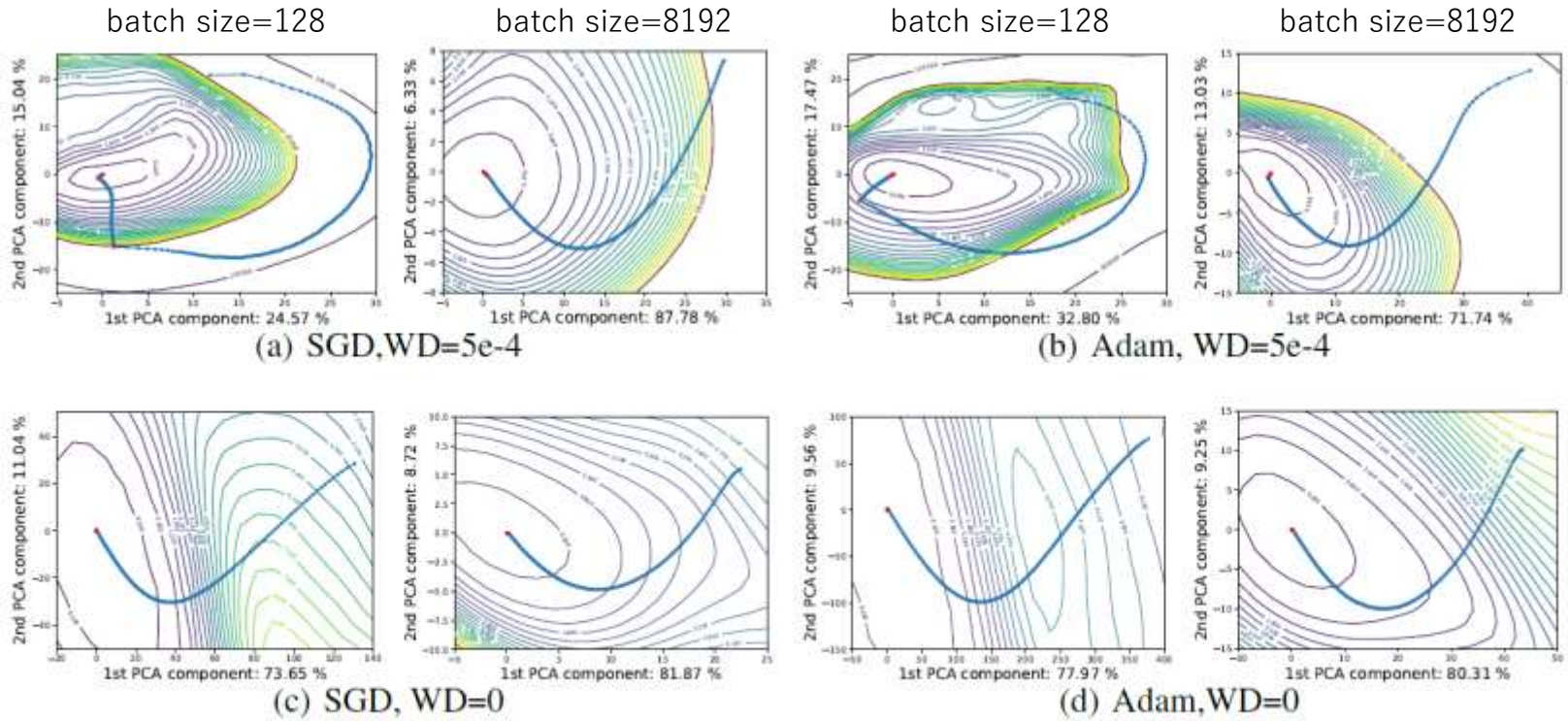


Figure 9: Projected learning trajectories use normalized PCA directions for VGG-9. The left plot in each subfigure uses batch size 128, and the right one uses batch size 8192.

*赤点でLearning Rateを下げている

おまけ

- Loss Landscapeというサイトで動画や画像が紹介されている
  - https://losslandscape.com/