Visualizing the Loss Landscape of Neural Nets

Hao Li[1] , Zheng Xu[1] , Gavin Taylor[2], Christoph Studer[3] , Tom Goldstein[1]
[1]University of Maryland, College Park, [2]United States Naval Academy, [3]Cornell University

https://arxiv.org/abs/1712.09913

Vikram Voleti | PhD, Mila

Presented at Mila, University of Montreal on 24th September, 2018

Contents

1. Contributions

VIsualizations have potential to answer-

- why are we able to minimize highly non-convex neural loss functions?
- And why do the resulting minima generalize?

Paper contributions:

- Reveal faults in other visualization methods
- Present simple visualization method based on “Filter Normalization”
- Observe that deeper architectures transit loss landscapes from convex to chaotic - coincides with decrease in generalization error
- Show that skip connections provide flat minima and prevent transit to chaos
- Visualize optimization trajectories

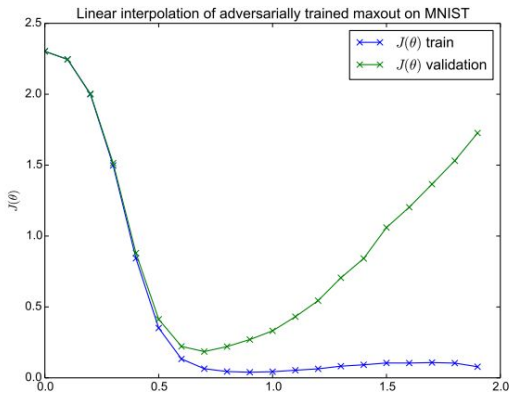
2. Past ways to visualize

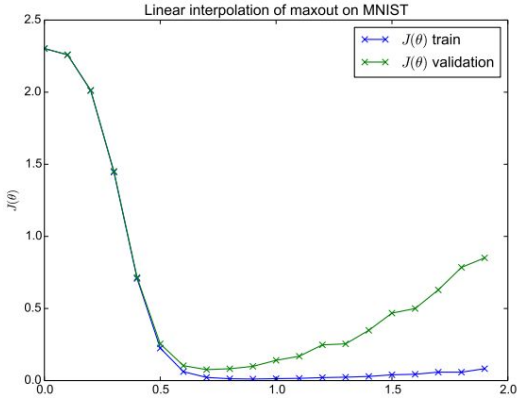
$$L(\theta) = \frac{1}{m} \sum_{i=1}^{m} \ell(x_i, y_i; \theta)$$

1D plot:

as seen in Goodfellow et al.:
https://arxiv.org/abs/1412.6544

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





Pros: used to study

- "sharpness" and "flatness" of different minima,
- dependence of sharpness on batch-size,
- different minima and the "peaks" between them,
- different minima obtained via different optimizers

Cons:

- difficult to visualize non-convexities,
- does not consider batch-norm or invariance symmetries in the network

2. Past ways to visualize

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

2D plot:

Choose centre point \theta-star, plot $f(\alpha, \beta)$ in random directions \delta and \eta

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

Pros: used to

- explore trajectories of different minimization methods
- show different optimization algorithms find different local minima

Cons:

- capture low-res plots of small regions,
- Non-convexities not captured

3. Proposal: Filter-Wise Normalization

Problem: **Scale Invariance**

- Multiply one ReLU output by 10 and divide at next layer by 10, output remains unchanged!
- Hence, comparisons of networks/optimizers might not be correct

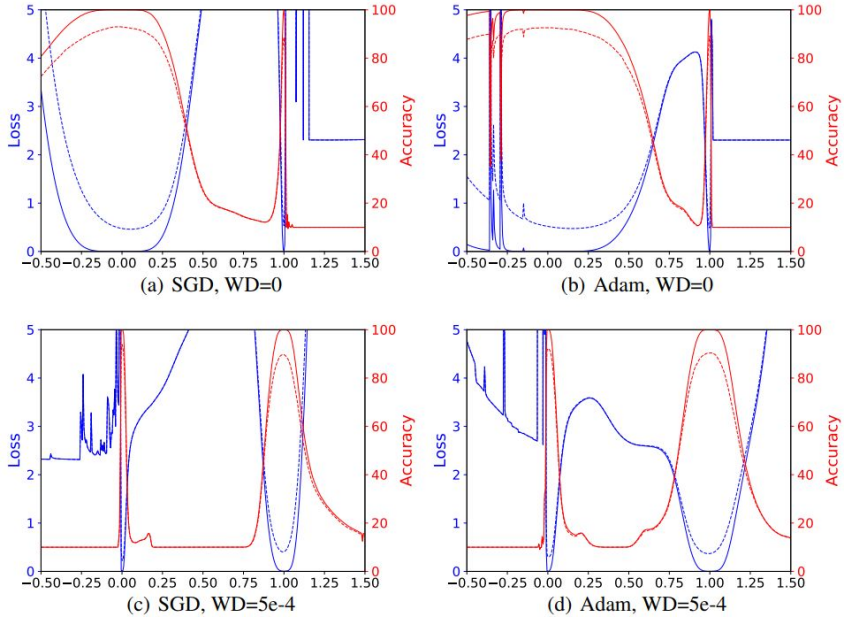
Solution: **Filter-wise Normalization**

- Generate random direction d from a Gaussian, then normalize using weights:

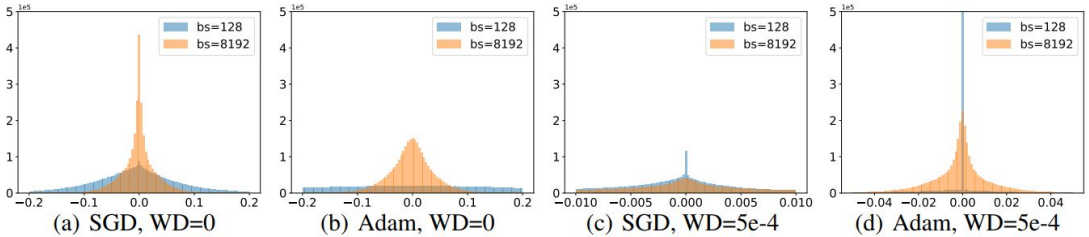
$$d_{i,j} \leftarrow \frac{d_{i,j}}{\|d_{i,j}\|} \|\theta_{i,j}\|$$

4. “Sharp” vs “Flat” minima

- **Unnormalized** plots show some correlation between small batch size and “sharp” minima w/o weight decay, and vice versa with



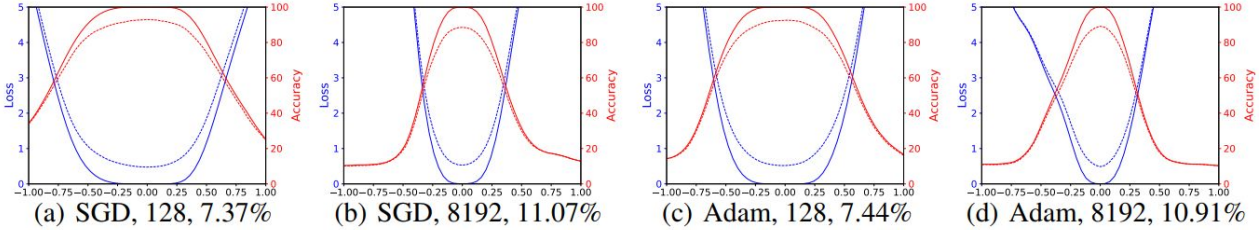
- But, can be correlated to weight values:



4. “Sharp” vs “Flat” minima

Filter-Wise Normalized Plots using random filter-normalized direction:

w/o Weight Decay:



with Weight Decay:

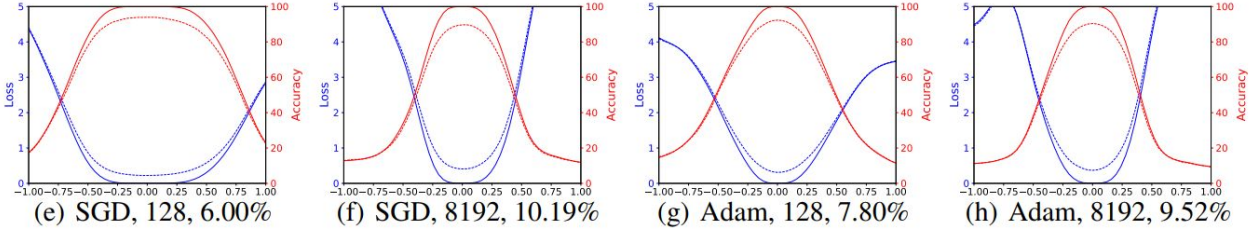


Figure 4: The shape of minima obtained using different optimization algorithms, batch size and weight decay. The title of each subfigure contains the optimizer, batch size, and test error. The first row has no weight decay and the second row uses weight decay 5e-4.

Conclusion: smaller batch-size => “flatter” minima, larger batch-size => “sharper” minima

4. “Sharp” vs “Flat” minima

Filter-Wise Normalized Plots using 2 random filter-normalized directions:

w/o Weight Decay:

with Weight Decay:

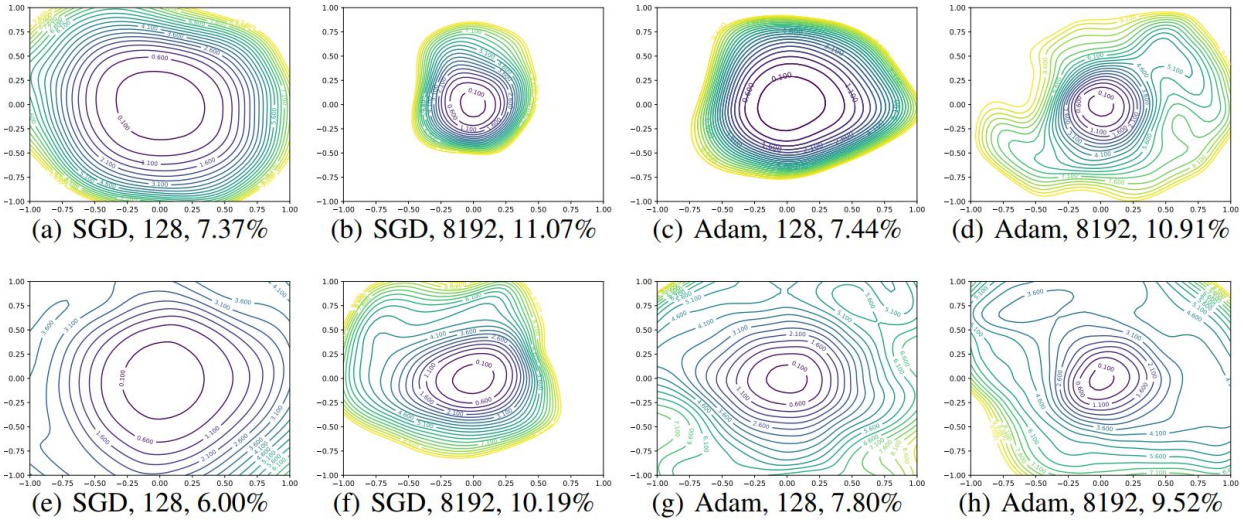


Figure 5: 2D visualization of solutions obtained by SGD with small-batch and large-batch. Similar to Figure 4, the first row uses zero weight decay and the second row sets weight decay to 5e-4.

Conclusion: the weights obtained with small batch size and non-zero weight decay have wider contours

4. “Sharp” vs “Flat” minima

Table 1: Test errors of VGG-9 on CIFAR-10 with different optimization algorithms and hyper-parameters.

	SGD		Adam	
	bs=128	bs=8192	bs=128	bs=8192
WD = 0	7.37	11.07	**7.44**	10.91
WD = 5e-4	**6.00**	**10.19**	7.80	**9.52**

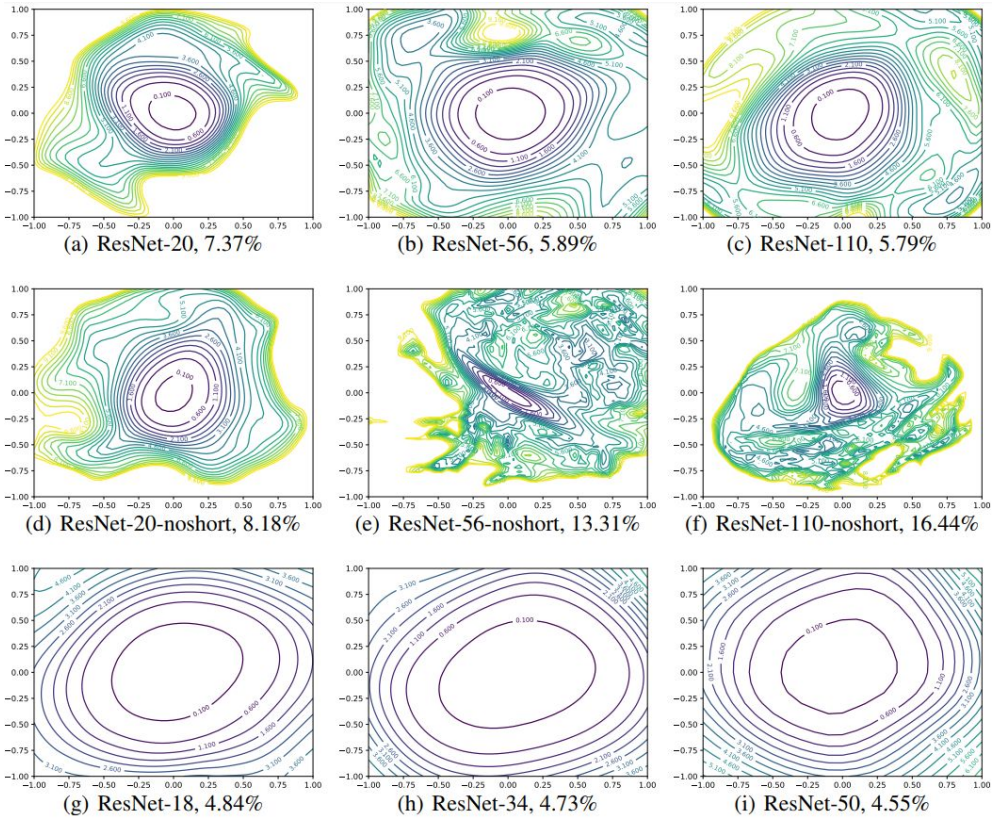
Conclusion: sharpness correlates well with generalization error

5. Minima of different architectures

Resnets
trained on CIFAR-10:

Resnets without skip connections
trained on CIFAR-10:

Wider Resnets
trained on ImageNet:



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

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

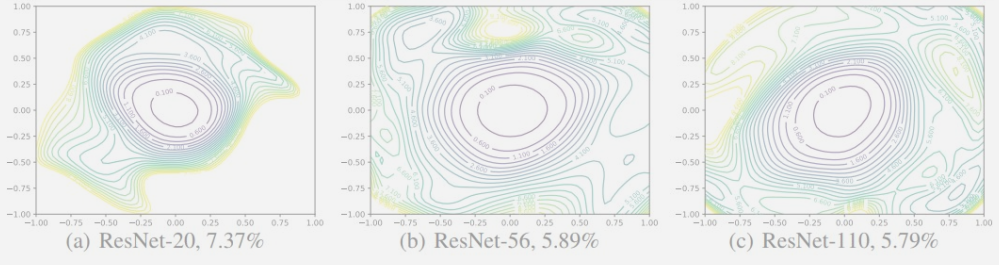
(g) ResNet-18, 4.84% (h) ResNet-34, 4.73% (i) ResNet-50, 4.55%

Figure 7: 2D visualization of the solutions of different networks.

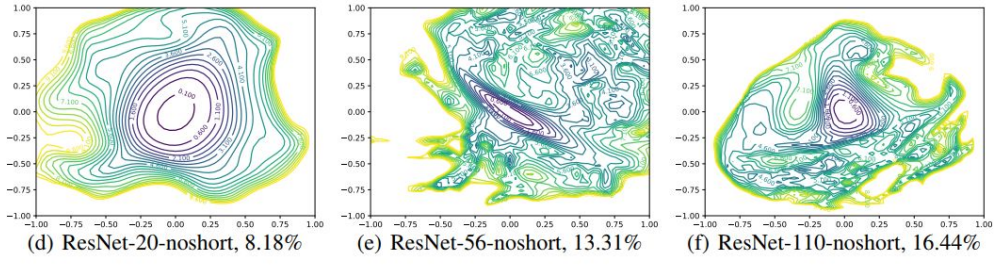
5. Minima of different architectures

DEPTH:

Resnets trained on CIFAR-10:



Resnets without skip connections trained on CIFAR-10:



Conclusion: depth makes convex->chaotic, extremely large in some directions, sharp minima, ill-conditioned with eccentricity

Wider Resnets trained on ImageNet:

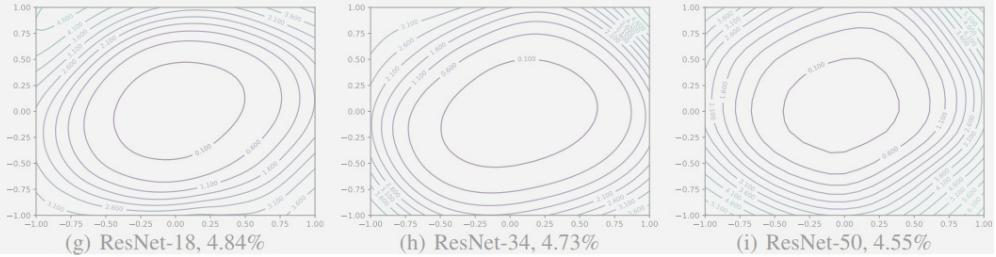


Figure 7: 2D visualization of the solutions of different networks.

5. Minima of different architectures

SKIP CONNECTIONS:

Resnets trained on CIFAR-10:

Conclusion: skip connections prevent transition to chaos with depth

Resnets without skip connections trained on CIFAR-10:

Wider Resnets trained on ImageNet:

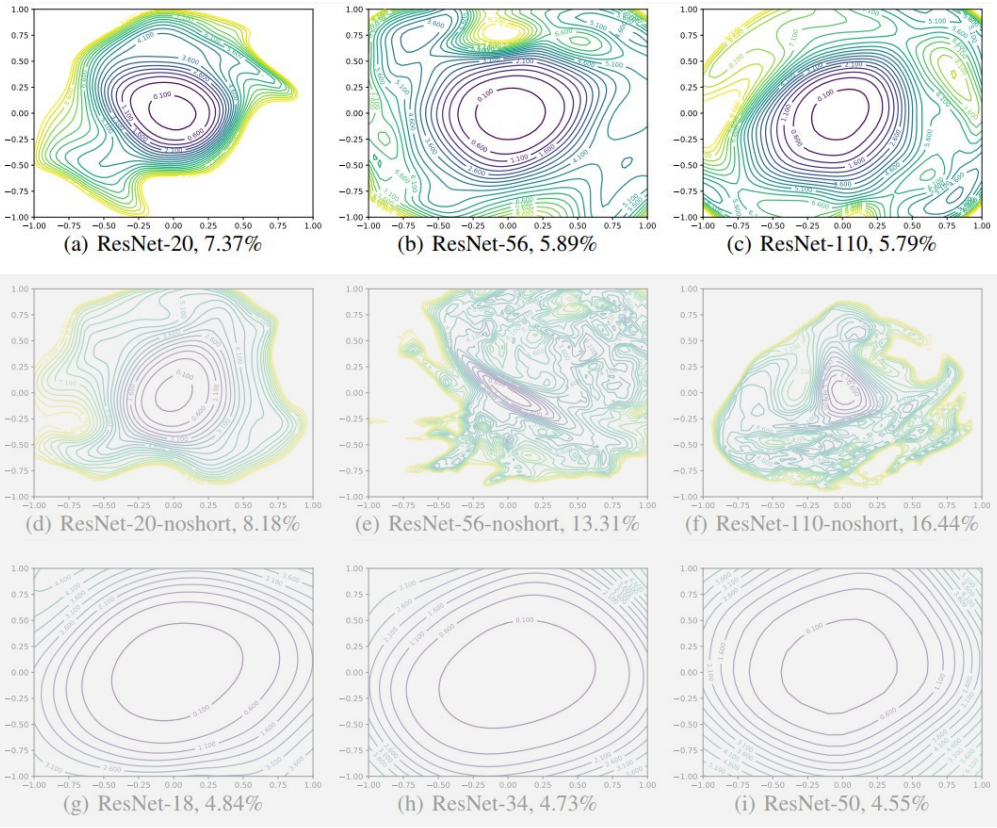
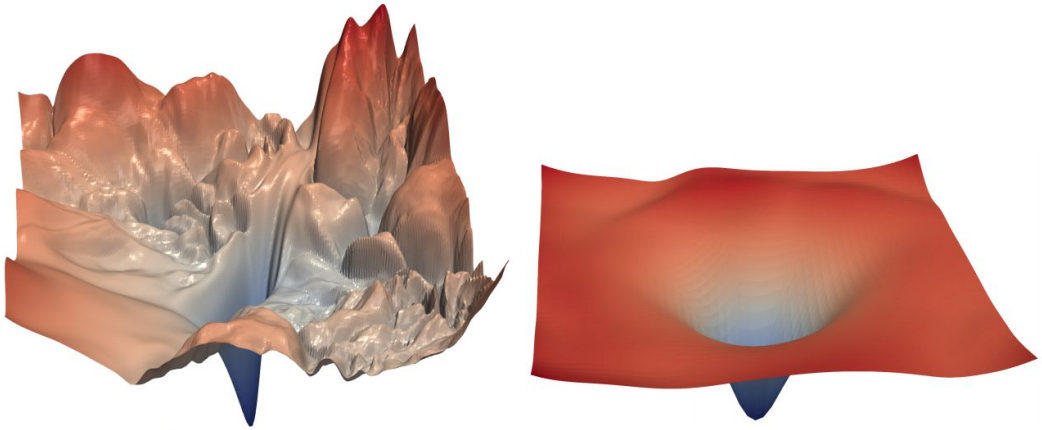


Figure 7: 2D visualization of the solutions of different networks.

5. Minima of different architectures

SKIP CONNECTIONS:



(a) without skip connections (b) with skip connections

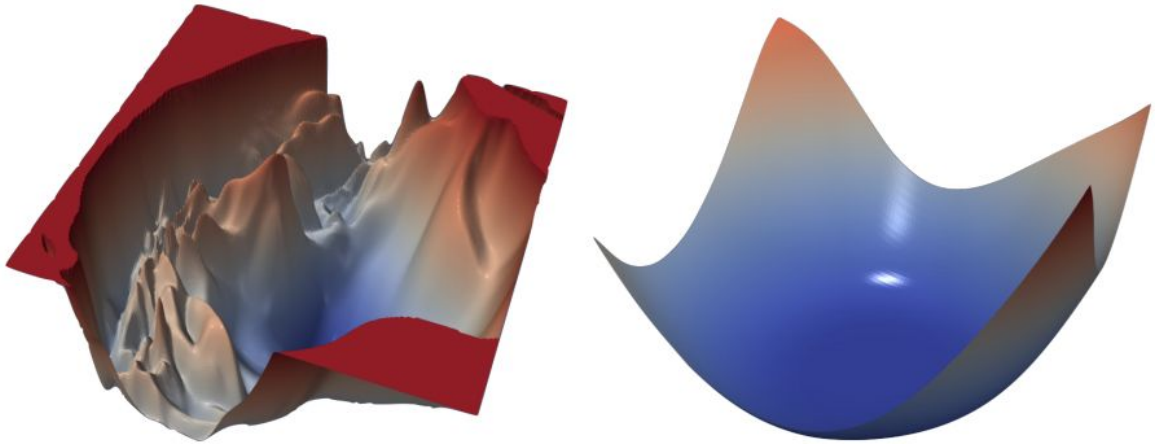
Figure 1: The loss surfaces of ResNet-56 with/without skip connections. The vertical axis is logarithmic to show dynamic range. The proposed filter normalization scheme is used to enable comparisons of sharpness/flatness between the two figures.

Conclusion: skip connections prevent transition to chaos with depth

Hmm, wonder how DenseNet's loss landscape is..

5. Minima of different architectures

SKIP CONNECTIONS:



(a) 110 layers, no skip connections (b) DenseNet, 121 layers

Figure 6: (left) The loss surfaces of ResNet-110-noshort, without skip connections. (right) DenseNet, the current state-of-the-art network for CIFAR-10.

Conclusion: skip connections prevent transition to chaos with depth

5. Minima of different architectures

WIDTH:

Resnets trained on CIFAR-10:

Resnets without skip connections trained on CIFAR-10:

Wider Resnets trained on ImageNet:

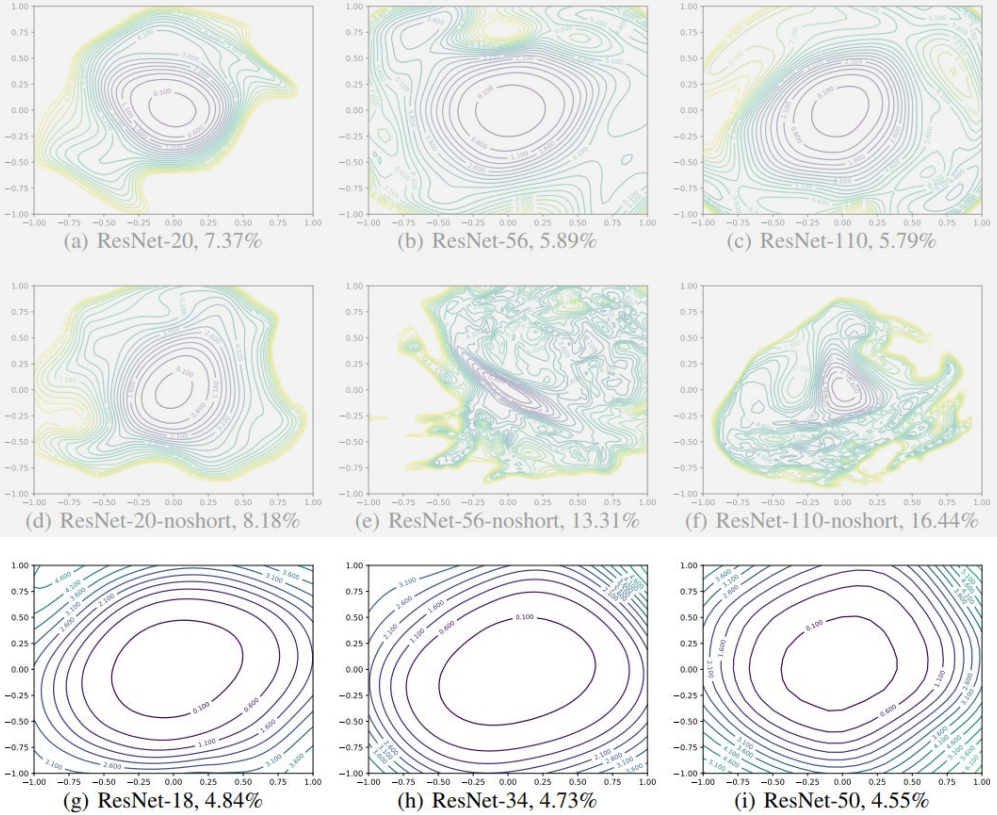
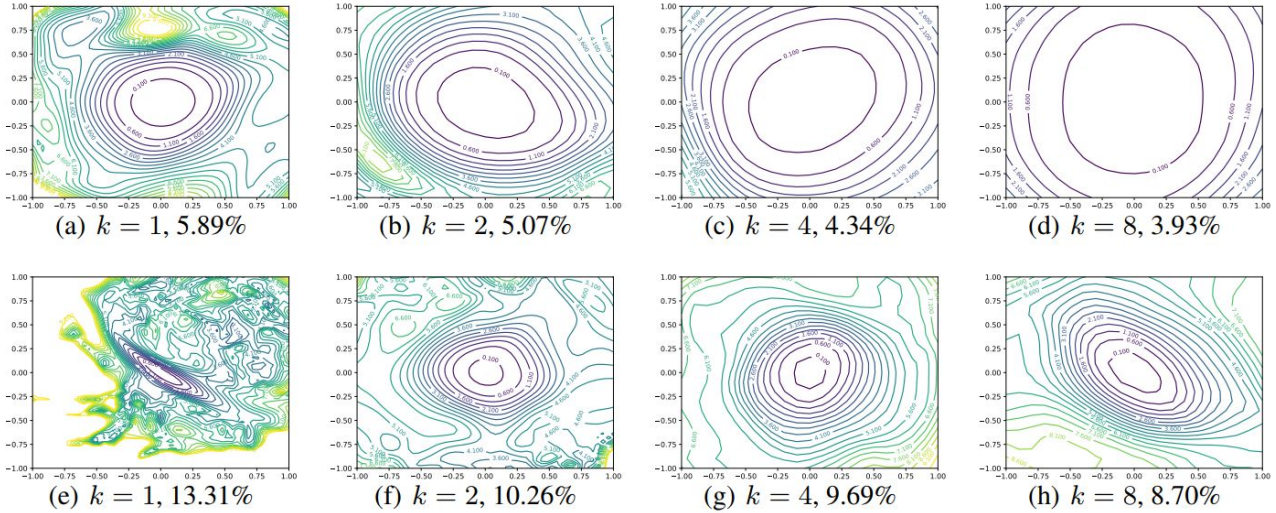


Figure 7: 2D visualization of the solutions of different networks.

Conclusion: wider (more filters per layer) networks have flatter minima and more convexity

5. Minima of different architectures

WIDTH:



(a) $k = 1$, 5.89% (b) $k = 2$, 5.07% (c) $k = 4$, 4.34% (d) $k = 8$, 3.93%

(e) $k = 1$, 13.31% (f) $k = 2$, 10.26% (g) $k = 4$, 9.69% (h) $k = 8$, 8.70%

Figure 8: Wide-ResNet-56 (WRN-56) on CIFAR-10 both with shortcut connections (top) and without (bottom). The label $k = 2$ means twice as many filters per layer, $k = 4$ means 4 times, etc. Test error is reported below each figure.

Conclusion: wider (more filters per layer) networks have flatter minima and more convexity

6. Visualizing optimization paths

Failed attempts:

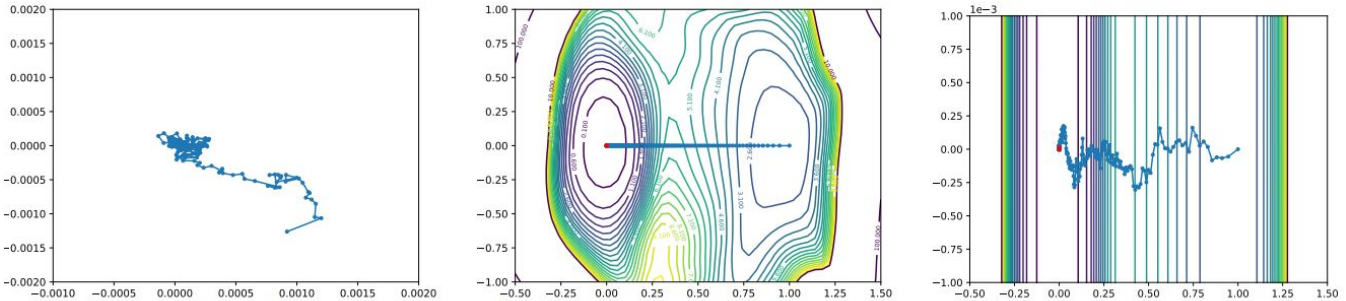


Figure 9: Ineffective visualizations of optimizer trajectories. These visualizations suffer from the orthogonality of random directions in high dimensions.

Conclusion: optimization path is highly low-dimensional! Can't pick random directions.

6. Visualizing optimization paths

PCA: Perform PCA on matrix M, and choose the best 2 directions:

$$M = [\theta_0 - \theta_n; \cdots; \theta_{n-1} - \theta_n]$$

(\theta_i is the weights at the ith epoch)

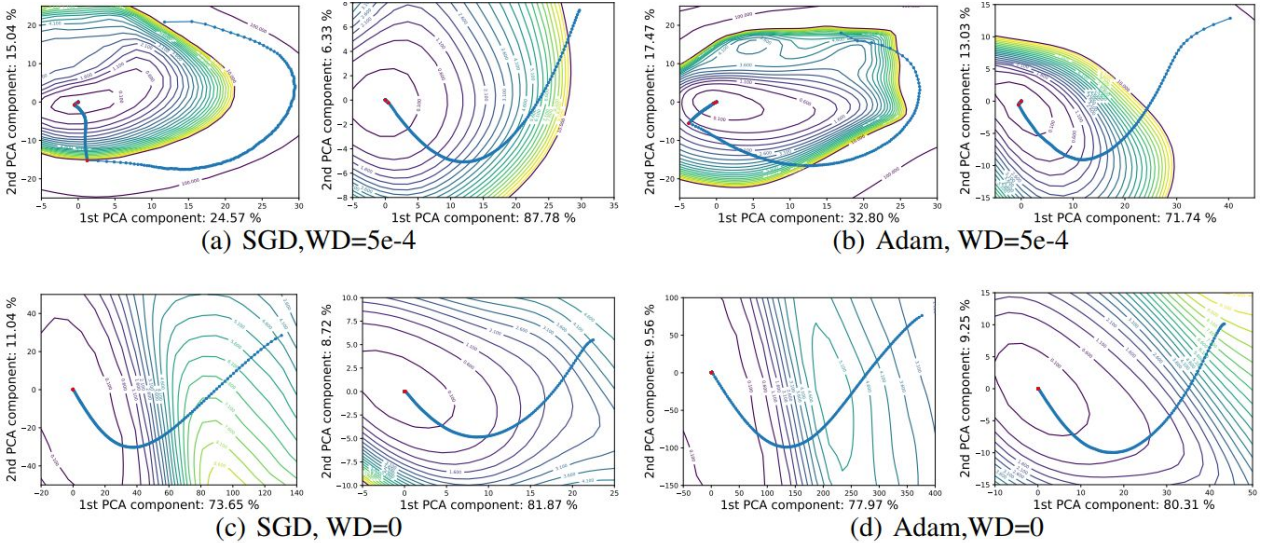


Figure 10: Projected learning trajectories use normalized PCA directions for VGG-9. The left plot in each subfigure uses batch size 128, and the right one uses batch size 8192.

7. Reviewer comments

- Proposed method is too incremental, not much novelty
- Feels preliminary, has potential
- Possibly only valid for ReLU, didn't compare for other activations

https://openreview.net/forum?id=HkmaTz-0W

Thank you.