Low-loss connection of weight vectors: distribution-based approaches

Ivan Anokhin, Dmitry Yarotsky

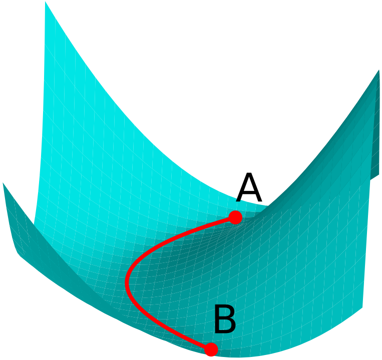
Skoltech
Skolkovo Institute of Science and Technology

ICML 2020

Introduction

How much connectedness is there in the bottom of a neural network's loss function?

Connection task: *Given two low-lying points (e.g., local minima), connect them by a possibly low lying curve.*



Low loss paths: existing approaches

Experimental [Garipov et al.'18, Draxler et al.'18]
Optimize the path numerically.

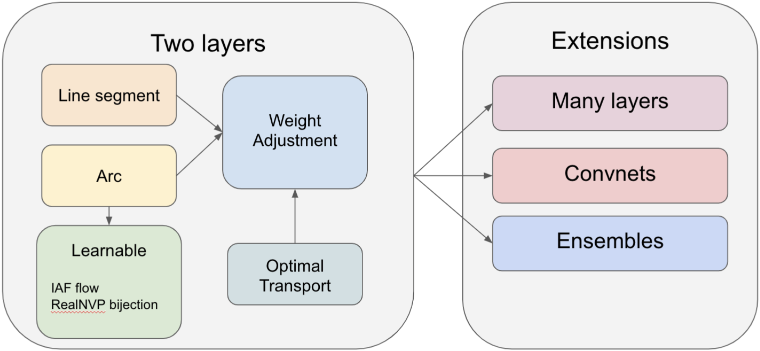
+ Generally applicable
+ Simple paths (e.g. two line segments)
– No explanation why it works

Theoretical [Freeman&Bruna'16, Nguyen'19, Kuditipudi et al.'19]
Prove existence of low loss paths.

+ Explain connectedness
– Relatively complex paths
– Require special assumptions on network

This work: a panel of methods

- Generally applicable
- Having a theoretical foundation
- Varying simplicity vs. performance (low loss)

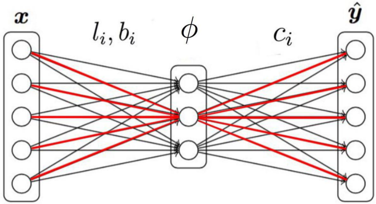


Two-layer network: the distributional point of view

Two-layer network:

$$\hat{\boldsymbol{y}}_n(\boldsymbol{x};\Theta) = \frac{1}{n}\sum_{i=1}^{n}\sigma(\boldsymbol{x};\boldsymbol{\theta}_i), \quad \Theta = (\boldsymbol{\theta}_i)_{i=1}^{n}$$

with $\boldsymbol{\theta_i} = (b_i, \boldsymbol{l}_i, \boldsymbol{c}_i)$ and $\sigma(x;\boldsymbol{\theta}_i) = \boldsymbol{c}_i\phi(\langle \boldsymbol{l}_i, x\rangle + b_i)$



Is an "ensemble of hidden neurons":

$$\hat{\boldsymbol{y}}_n(\boldsymbol{x};\Theta) = \int \sigma(\boldsymbol{x};\boldsymbol{\theta})p(d\boldsymbol{\theta})$$

with distribution $p = \frac{1}{n}\sum_{i=1}^{n}\delta_{\boldsymbol{\theta}_i}$

Connection by distribution-preserving paths

Key assumption: networks A and B trained under similar conditions have approximately the same distribution p of their hidden neurons $\boldsymbol{\theta}_i^A, \boldsymbol{\theta}_i^B$.

Choose connection path $\Psi(t) = (\psi_i(t))$ so that

1. For each i, $\psi_i(t=0) = \boldsymbol{\theta}_i^A$ and $\psi_i(t=1) = \boldsymbol{\theta}_i^B$
2. For each t, $\psi(t) \sim p$

Then the network output is approximately t-independent, and loss is constant

Linear connection

The simplest possible connection:

$$\psi(t) = (1-t)\boldsymbol{\theta}^A + t\boldsymbol{\theta}^B$$

+ If $\boldsymbol{\theta}^A, \boldsymbol{\theta}^B \sim p$, then $\psi(t)$ preserves the mean $\boldsymbol{\mu} = \int \boldsymbol{\theta} dp$
− $\psi(t)$ *does not* preserve covariance $\int (\boldsymbol{\theta} - \boldsymbol{\mu})(\boldsymbol{\theta} - \boldsymbol{\mu})^T dp$

The Gaussian-preserving flow

Proposition

If $\boldsymbol{\theta}^A, \boldsymbol{\theta}^B$ *are i.i.d. vectors with the same centered multivariate Gaussian distribution, then for any* $t \in \mathbb{R}$

$$\psi(t) = \cos(\tfrac{\pi}{2}t)\boldsymbol{\theta}^A + \sin(\tfrac{\pi}{2}t)\boldsymbol{\theta}^B$$

has the same distribution, and also $\psi(0) = \boldsymbol{\theta}^A, \psi(1) = \boldsymbol{\theta}^B$

Arc connection

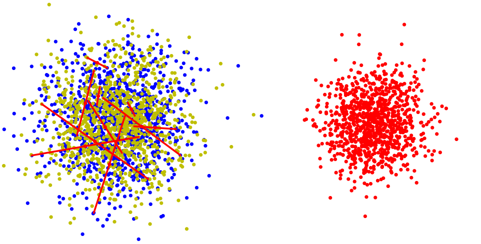
$$\psi(t) = \boldsymbol{\mu} + \cos(\tfrac{\pi}{2}t)(\boldsymbol{\theta}^A - \boldsymbol{\mu}) + \sin(\tfrac{\pi}{2}t)(\boldsymbol{\theta}^B - \boldsymbol{\mu})$$

+ Preserves shifted Gaussian p with mean $\boldsymbol{\mu}$
+ For a general non-Gaussian p with mean $\boldsymbol{\mu}$, preserves mean and covariance of p

Linear and Arc connections

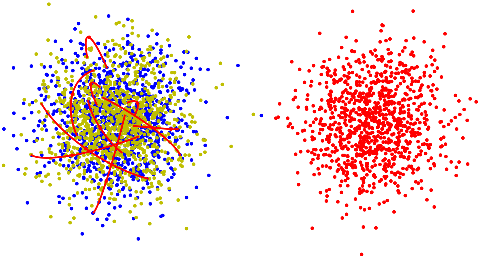
Connected distributions | Middle of path

Linear: distribution "squeezed"



X, Y | $0.5X + 0.5Y$

Arc: distribution preserved



X, Y | $\cos(\pi/4)X + \sin(\pi/4)Y$

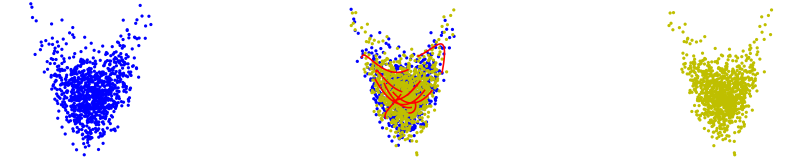
Distribution-preserving deformations: general p

For a general non-Gaussian distribution p, if ν maps p to $\mathcal{N}(0, I)$, then the path

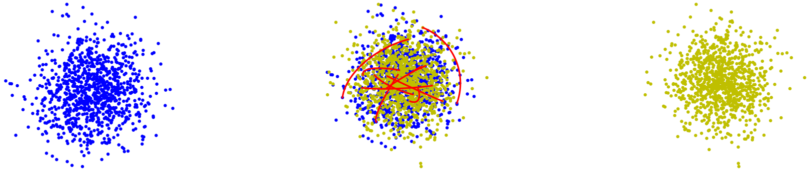
$$\psi(t) = \nu^{-1}[\cos(\tfrac{\pi}{2}t)\nu(\boldsymbol{\theta}^A) + \sin(\tfrac{\pi}{2}t)\nu(\boldsymbol{\theta}^B)]$$

is p-preserving

Connections using a normalizing map



$$
\begin{tikzcd}
\boldsymbol{\theta}^A \arrow[d, "\nu"'] & \psi(t) & \boldsymbol{\theta}^B \arrow[d, "\nu"'] \\
\widetilde{\boldsymbol{\theta}}^A_{normal} \arrow[r] & \cos(\frac{\pi}{2}t)\widetilde{\boldsymbol{\theta}}^A_{normal} + \sin(\frac{\pi}{2}t)\widetilde{\boldsymbol{\theta}}^B_{normal} \arrow[u, "\nu^{-1}"] & \widetilde{\boldsymbol{\theta}}^B_{normal} \arrow[l]
\end{tikzcd}
$$



Flow connection

Learn ν to map from target distribution p to $\mathcal{N}(0, I)$ by using *Normalizing Flow* [Dinh et al.'16, Kingma et al.'16]:

$$\mathbb{E}_{\boldsymbol{\theta} \sim p} \log \left[\rho(\nu(\boldsymbol{\theta})) \middle| \det \frac{\partial \nu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^T} \middle| \right] \to \max_{\nu},$$

where ρ is the density of $\mathcal{N}(0, I)$

Bijection connection

$$\psi_W(t, \Theta^A, \Theta^B) = \nu_W^{-1}[\cos(\tfrac{\pi}{2}t)\nu_W(\Theta^A) + \sin(\tfrac{\pi}{2}t)\nu_W(\Theta^B)]$$

Train ν_W to have low-loss path between any optima, Θ^A and Θ^B, with loss

$$l(W) = \mathbb{E}_{t\sim U(0,1),\Theta^A\sim p,\Theta^B\sim p} L(\psi_W(t, \Theta^A, \Theta^B)),$$

where $L(W)$ is the initial loss with which we train the models Θ^A and Θ^B

Learnable connection methods

For both **Flow** and **Bijection** connections:

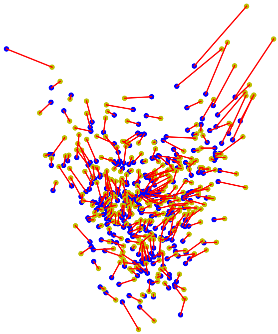
- We train learnable connection methods using a dataset of trained model weights Θ;
- We use the networks RealNVP [Dinh et al.'16] and IAF [Kingma et al.'16] as ν-transforms.

The result is a *global connection model*: once trained, it can be applied to any pair of local minima Θ^A, Θ^B

Connection using Optimal Transportation (OT)

Stage 1: connect $\{\boldsymbol{\theta}_i^A\}_{i=1}^n$ to $\{\boldsymbol{\theta}_i^B\}_{i=1}^n$ as *unordered sets*

- Use OT to find a bijective map from samples $\boldsymbol{\theta}_i^A$ to nearby samples $\boldsymbol{\theta}_{\pi(i)}^B$
- Interpolate linearly between respective samples



Stage 2: permute the neurons one-by-one to get the right order

Connections using Weight Adjustment (WA)

A two-layer network: $\mathbf{Y} = W_2\phi(W_1\mathbf{X})$

Given two two-layer networks, A and B:

- Connect the first layers $W_1(t) = \psi(t, W_1^A, W_1^B)$ with any considered connection method (e.g. **Linear, Arc, OT**).
- Adjust the second layer by pseudo-inversion to keep the output possibly t-independent: $W_2(t) = \mathbf{Y}\left[\phi(W_1(t)\mathbf{X})\right]^+$

We consider: Linear + WA, Arc + WA and OT + WA.

Overview of the methods

	Explicit formula	Learnable	Compute resources	Path complexity	Loss on path
Linear	+	–	low	low	high
Arc	+	–	low	low	high
Flow	–	+	medium	medium	high
Bijection	–	+	medium	medium	low
OT	–	–	medium	high	low
WA based	–	–	high	high	low

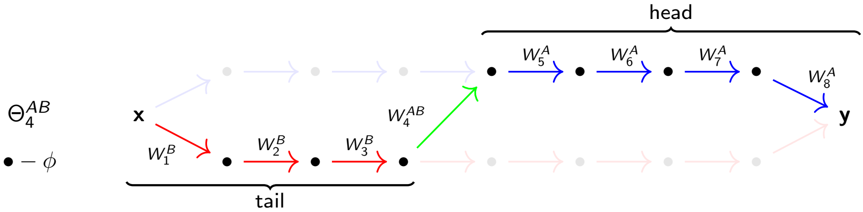
Experiments (two layer networks)

The worst accuracy (%) along the path for networks with 2000 hidden ReLU units

	MNIST		CIFAR10	
Methods	train	test	train	test
Linear	96.54 ± 0.40	95.87 ± 0.40	32.09 ± 1.33	39.34 ± 1.52
Arc	97.89 ± 0.11	97.03 ± 0.14	49.97 ± 0.86	41.34 ± 1.39
IAF flow	96.34 ± 0.54	95.80 ± 0.45	–	–
RealNVP bijection	98.50 ± 0.09	97.53 ± 0.11	63.46 ± 0.27	53.94 ± 0.95
Linear + WA	98.76 ± 0.01	97.86 ± 0.05	52.63 ± 0.59	57.66 ± 0.26
Arc + WA	98.75 ± 0.01	97.86 ± 0.05	58.77 ± 0.32	57.88 ± 0.24
OT	98.78 ± 0.01	97.87 ± 0.04	66.19 ± 0.23	56.49 ± 0.46
OT + WA	98.92 ± 0.01	97.91 ± 0.03	67.02 ± 0.12	58.96 ± 0.21
Garipov (3)	99.10 ± 0.01	97.98 ± 0.02	68.51 ± 0.08	58.74 ± 0.23
Garipov (5)	99.03 ± 0.01	97.93 ± 0.02	67.20 ± 0.12	57.88 ± 0.32
End Points	99.14 ± 0.01	98.01 ± 0.03	70.60 ± 0.12	59.12 ± 0.26

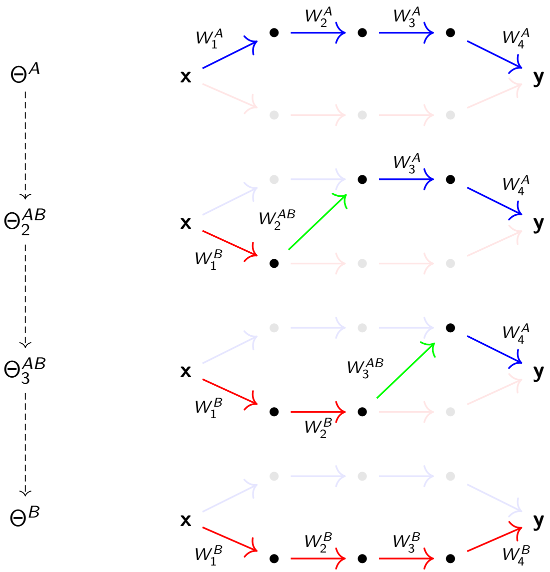
Connection of multi layer networks

An intermediate point Θ_k^{AB} on the path has head of network A attached to tail of network B



We adjust the transitional layer W_k^{AB} using the Weight Adjustment procedure, to preserve the output of the k'th layer of network A

The full path: $\Theta^A \to \Theta_2^{AB} \to \Theta_3^{AB} \to \cdots \to \Theta_n^{AB} \to \Theta^B$



The transition $\Theta_k^{AB} \to \Theta_{k+1}^{AB}$

- Θ_k^{AB} and Θ_{k+1}^{AB} differ only in layers k and $k+1$
- Connect Θ_k^{AB} to Θ_{k+1}^{AB} like a two-layer network

Experiments. Three layer MLP

The worst accuracy (%) along the path for networks with 6144 and 2000 hidden ReLU units

	CIFAR10	
Methods	train	test
Linear	47.81 ± 0.76	38.38 ± 0.84
Arc	60.60 ± 0.79	49.63 ± 0.86
Linear + WA	60.93 ± 0.25	51.87 ± 0.24
Arc + WA	71.10 ± 0.23	58.86 ± 0.29
OT	81.95 ± 0.29	59.11 ± 0.46
OT + WA	87.53 ± 0.18	61.67 ± 0.49
Garipov (3)	94.56 ± 0.08	61.38 ± 0.36
Garipov (5)	90.32 ± 0.06	60.75 ± 0.32
End Points	95.13 ± 0.08	63.25 ± 0.36

Convnets

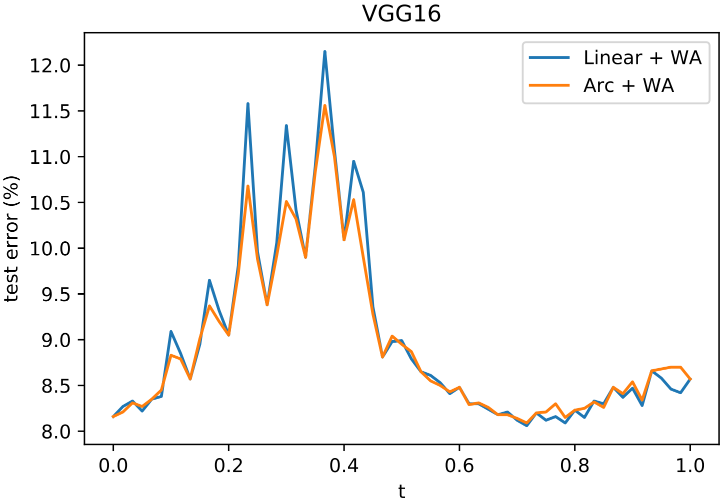
For CNNs, connection methods work similarly to dense nets, but with filters instead of neurons

	Conv2FC1		VGG16	
Methods	train	test	train	test
Linear + WA	71.09 ± 0.38	67.07 ± 0.49	94.16 ± 0.38	87.55 ± 0.41
Arc + WA	77.36 ± 0.99	73.77 ± 0.88	95.35 ± 0.23	88.56 ± 0.28
Garipov (3)	85.10 ± 0.25	80.95 ± 0.16	99.69 ± 0.03	91.25 ± 0.14
End Points	87.18 ± 0.14	82.61 ± 0.18	$99.99 \pm 0.$	91.67 ± 0.10

Accuracy (%) of three layer convnet, Conv2FC1 and VGG16, on CIFAR10.
Conv2FC1 has 32 and 64 channels in convolution layers and ~ 3000 neurons in FC

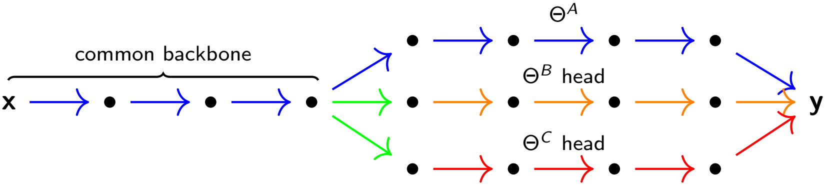
Experiments. VGG16

Test error (%) along the path for VGG16



WA-Ensembles

- Take m independently trained networks $\Theta^A, \Theta^B, \Theta^C, \ldots$
- Take the tail of network Θ^A up to some layer k as a backbone;
- Use WA to transform the other networks to have the same backbone;
- Make ensemble with the common backbone.



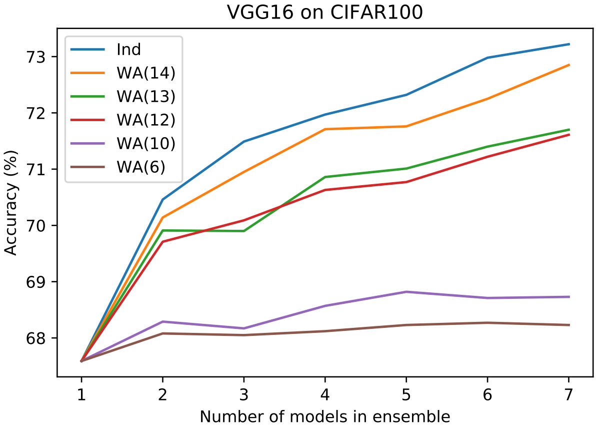
Compared to the usual ensemble:

+ Smaller storage & complexity (thanks to common backbone);
– Lower accuracy (due to errors introduced by WA).

Experiments. WA-Ensembles. VGG16

Test accuracy (%) of ensemble methods with respect to number of models.

- **WA(n)**: WA-ensemble with n layers in the head
- **Ind**: usual ensemble – averaging of independent models ($\equiv$ WA(16))



Take away

- Simple **Arc** modification noticeably improves the trivial **Linear** connection.
- **Optimal Transportation** with **Weight Adjustment** based connection method achieves low loss on par with direct numerical optimization, but is more interpretable.
- In **WA-ensembles**, a longer common backbone reduces amount of computation at the cost of accuracy.