How to Guess a Gradient

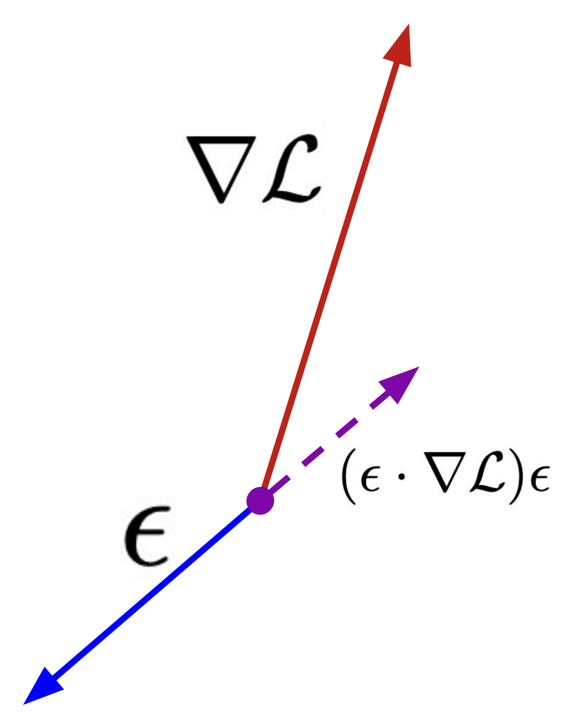
Utkarsh **Singhal** | Brian **Cheung** | Kartik **Chandra** | Jonathan **Ragan-Kelley** | Joshua B **Tenenbaum** | Tomaso **Poggio** | Stella **Yu**

Summary: *We guess a neural network's gradients without computing a loss or knowing the label*

Backpropagation requires a lot of memory and is not biologically plausible. ***Directional descent*** *is a previously proposed alternative:*

1. *Pick a random direction* **ε**
2. *Find directional derivative along* **ε** *using forward-mode automatic differentiation (cheap!).*
3. *Scale* **ε** *by directional derivative:*

$$w_{t+1} = w_t - \alpha(\epsilon \cdot \nabla\mathcal{L})\epsilon$$

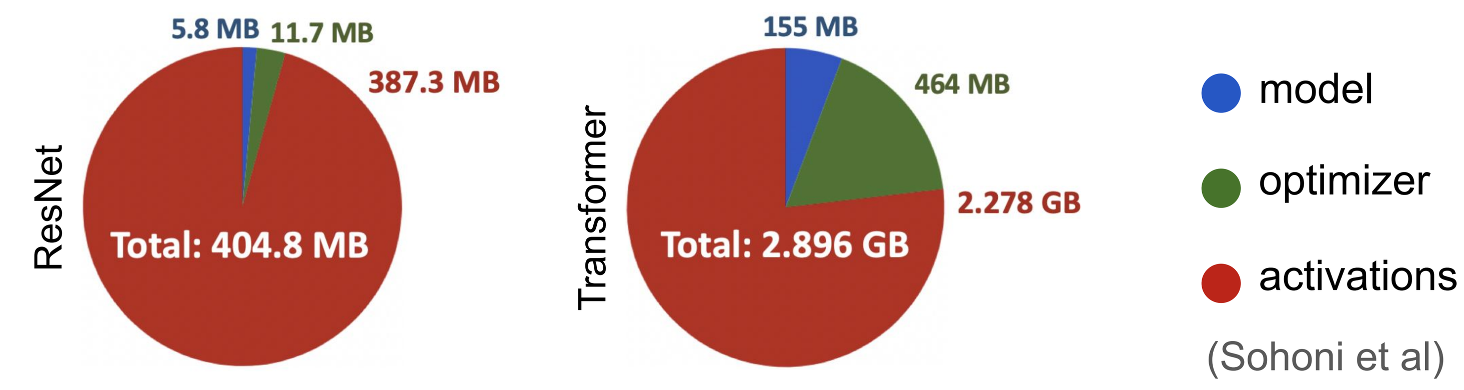


Pros:
- *Unbiased estimator of* $\nabla\mathcal{L}$
- *Guaranteed to be within 90° of true gradient.*
- *Doesn't require storing activations like backprop*

Cons:
- *Cosine similarity decreases with guess dimensionality as* $O(\frac{1}{\sqrt{N}})$

Main Question: *How can we narrow guess space?*

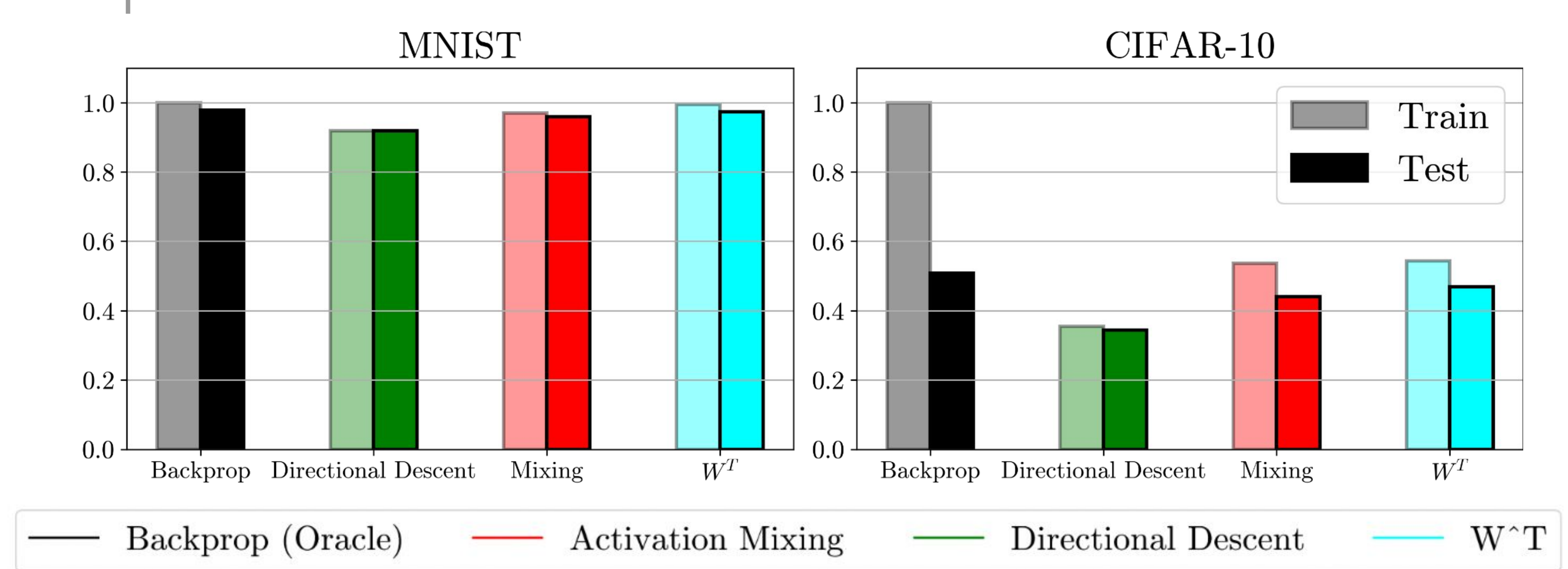
Answer: *Use local feature/architecture knowledge!*



Results – MLPs

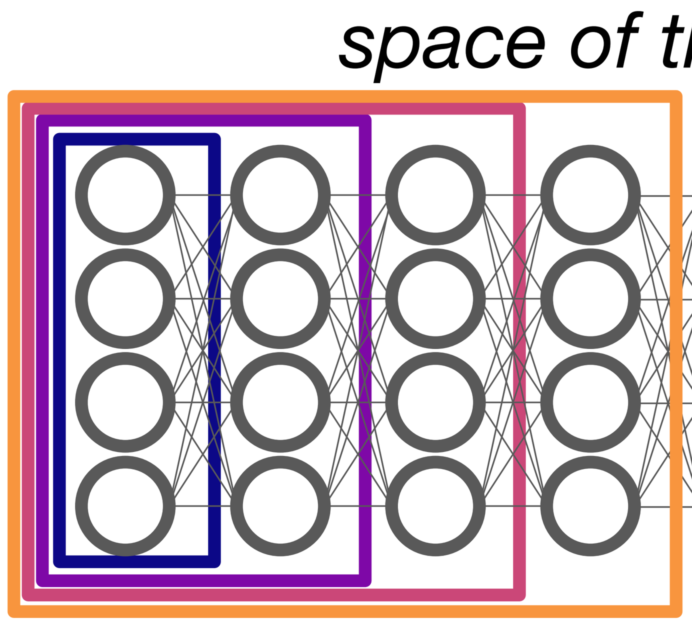
Directional Descent*: Random guess for weights*
Activation Perturbation*: Random guess for activations*

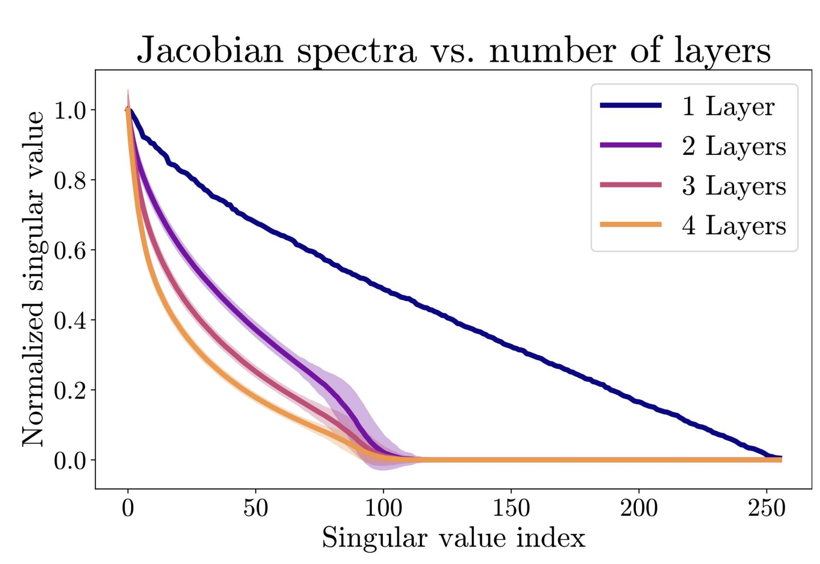
	Method	Cosine Similarity	1-step effectiveness
	Backprop (Oracle)	1	1
	Directional Descent	0.0003	1×10^{-6}
Ours	Activation Perturbation	0.016	6.9×10^{-4}
Ours	Activation Mixing	0.025	$\mathbf{3.4 \times 10^{-3}}$
Ours	W^T	**0.030**	1.7×10^{-3}



Architecture-based guessing

Observation: *gradients lie in the column space of the Jacobian matrix*





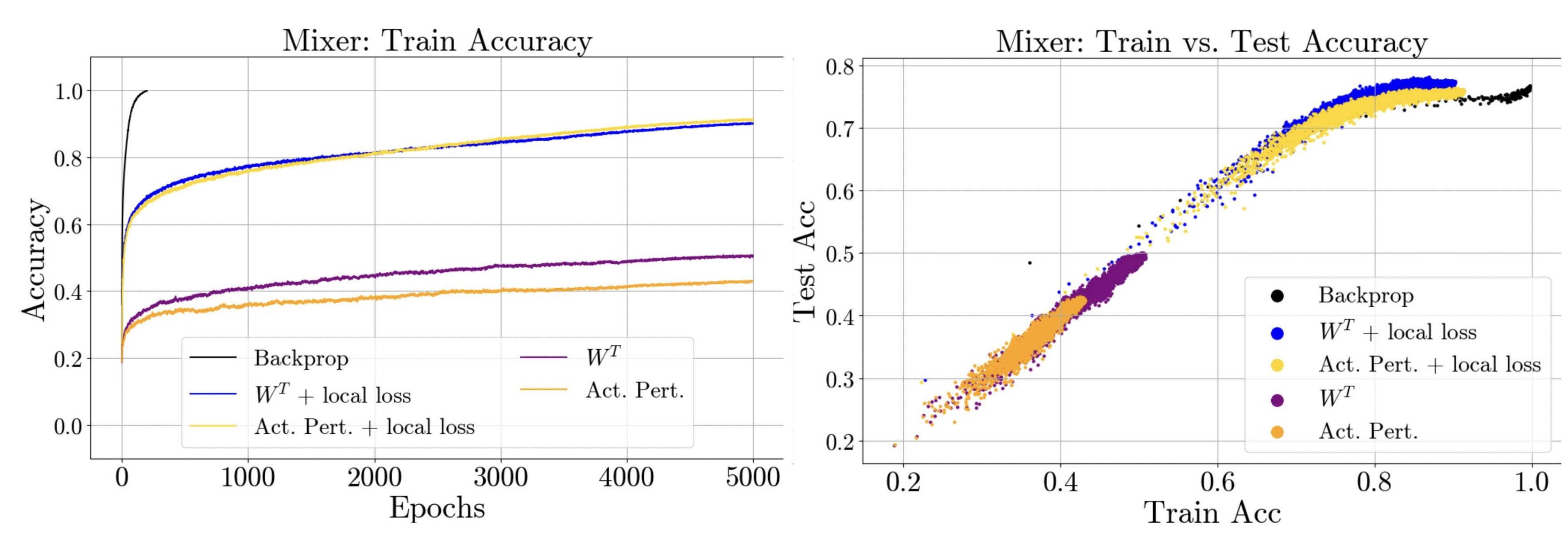
Idea (W^T): *Filter a random guess through the next layer's weight matrix (part of Jacobian)*

$$g_{true} = J_1 J_2 J_3 \dots J_n e$$

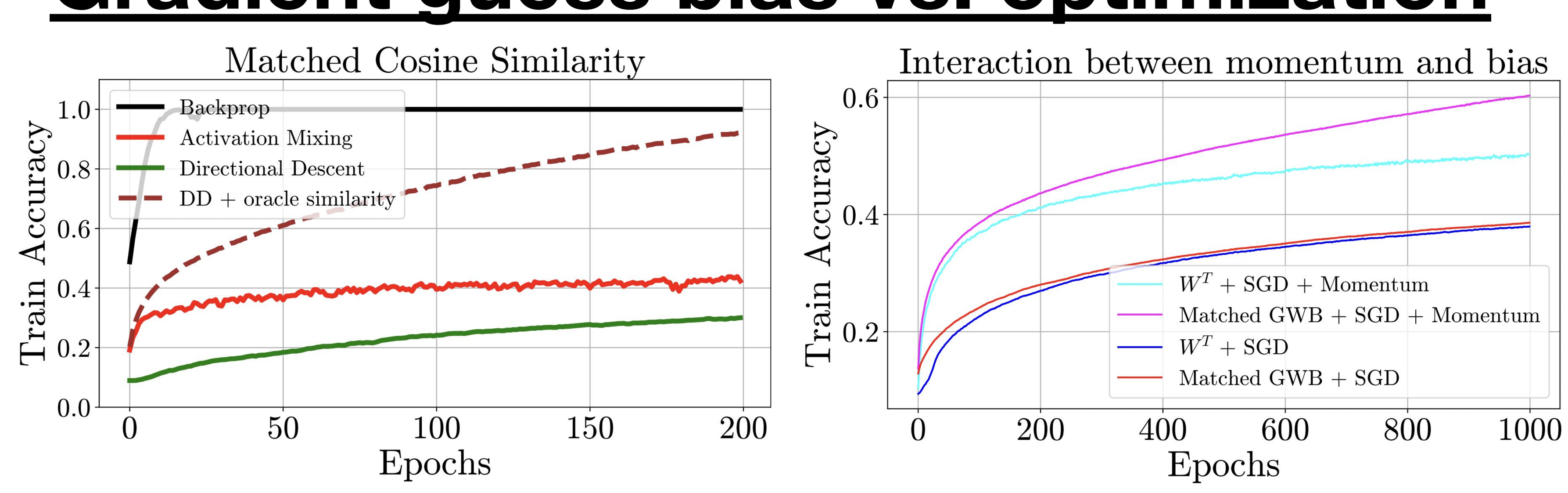
$$\epsilon = J_1 \eta, \quad \eta \sim \mathcal{N}(0, I)$$

Results – LocalMixer

	Backprop	Ren et al. (2022)	Mixing	W^T
Reported (Ren et al. (2022))	66.4	**69.3**	-	-
Reproduced with Adam	71.2	71.2	68.8	**72.5** (+1.3)
Augmentation (500 epochs)	**76.4**	72.2	68.2	74.4 (+1.2)
Augmentation (5000 epochs)	**77.6**	76	69.4	77.4 (+1.4)



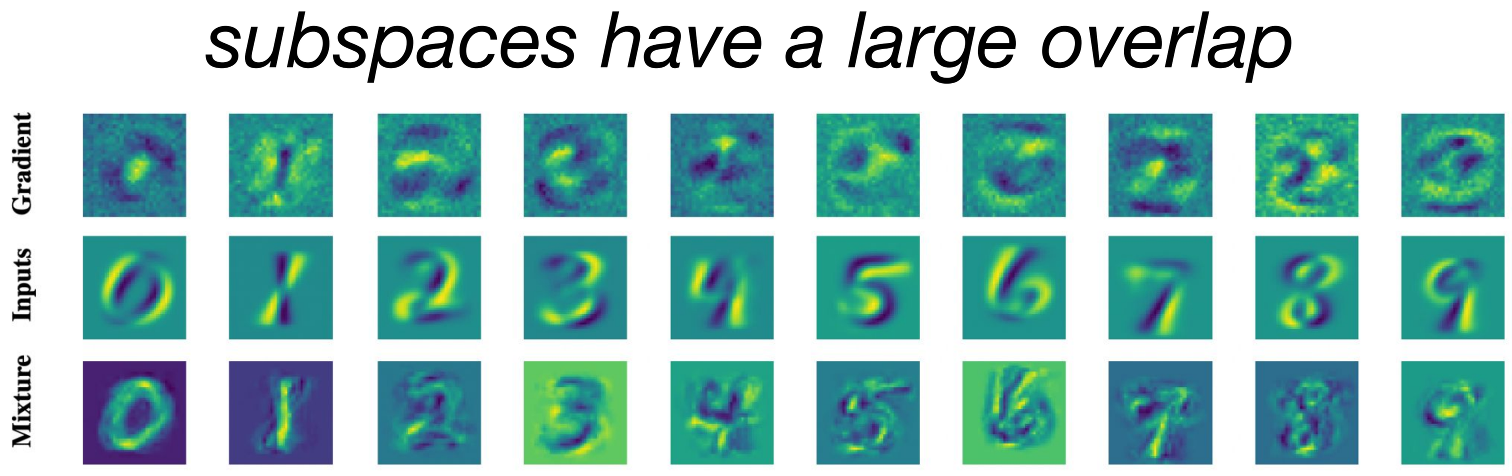
Gradient guess bias vs. optimization

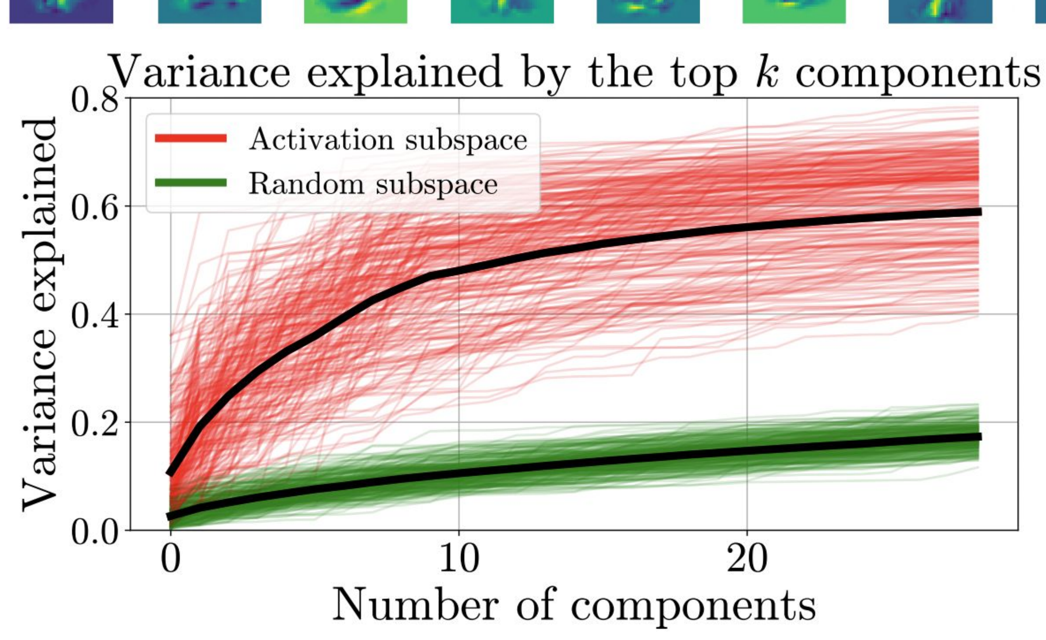


Bias impedes convergence when momentum is used

Data-based guessing

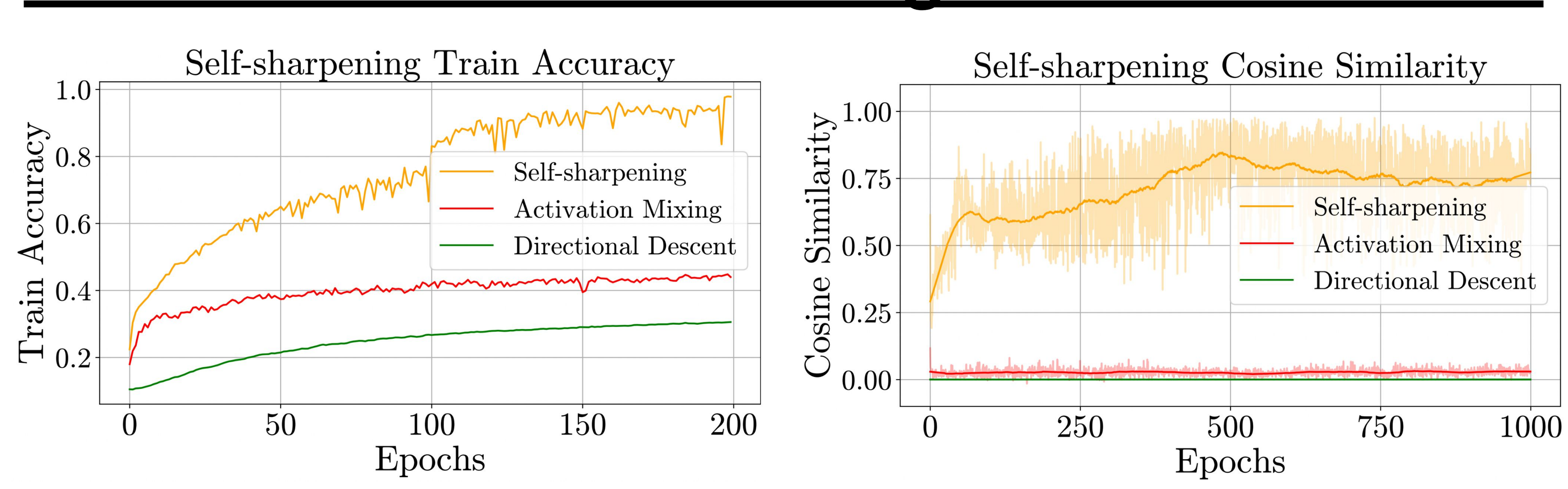
Observation: *gradient and activation subspaces have a large overlap*





Idea (Activation Mixing): *Use random mixture of activations as guess*

How bias leads to better guesses over time



Low-rank guess→ low-rank weight → smaller guess space → higher cosine similarity