LSDL Lecture 06
Ensembling and Weight Averaging

Ildus Sadrtdinov, 21.10.24

Plan

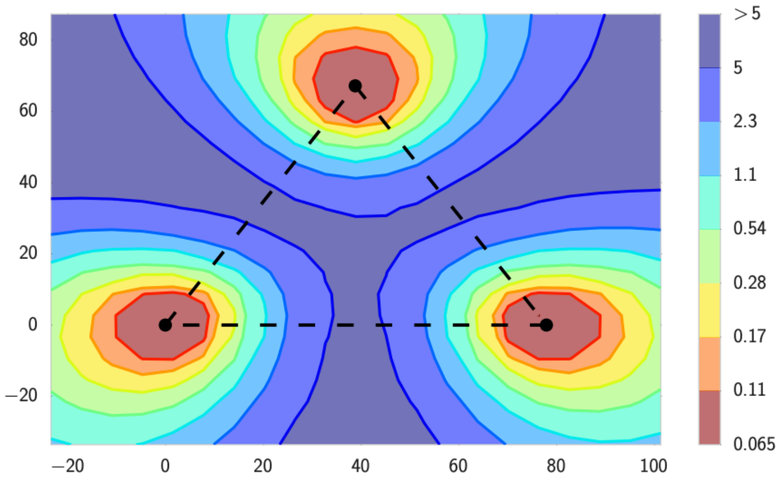
- Mode connectivity and Linear Mode Connectivity (LMC)
- Weight averaging techniques
  - Stochastic Weight Averaging (SWA)
  - Model soups
- Ensembling methods
  - Deep Ensembles
  - Cyclical methods: SSE, FGE, cSGLD
  - Non-cyclical methods: KFAC-Laplace, SWAG, SPRO
- Ensembles in transfer learning
- Weight-averaging based optimizers

Plan

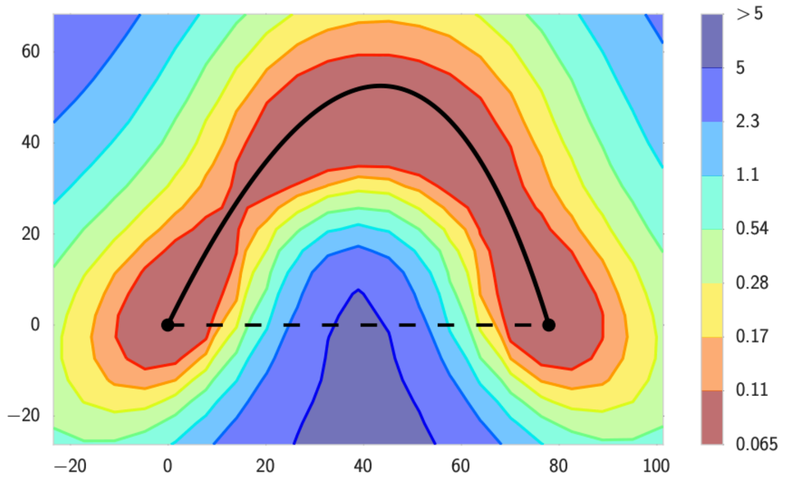
- **Mode connectivity and Linear Mode Connectivity (LMC)**
- Weight averaging techniques
  - Stochastic Weight Averaging (SWA)
  - Model soups
- Ensembling methods
  - Deep Ensembles
  - Cyclical methods: SSE, FGE, cSGLD
  - Non-cyclical methods: KFAC-Laplace, SWAG, SPRO
- Ensembles in transfer learning
- Weight-averaging based optimizers

Mode connectivity

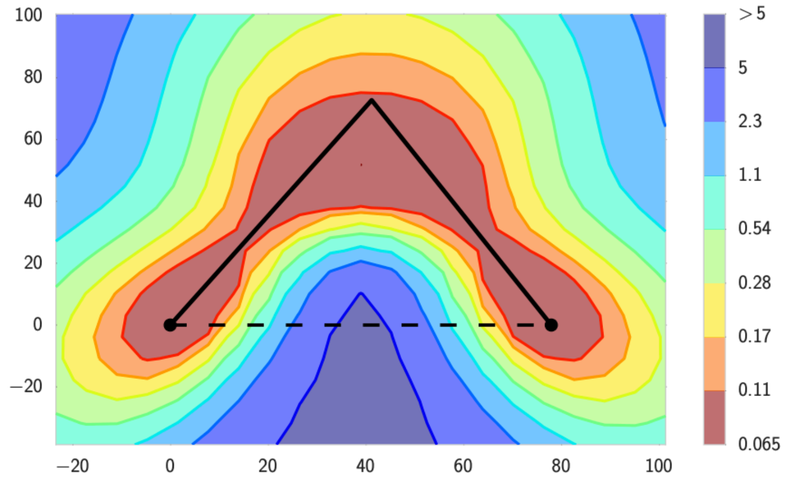
Three independently trained networks



Bezier connecting curve



Polygonal chain connecting curve

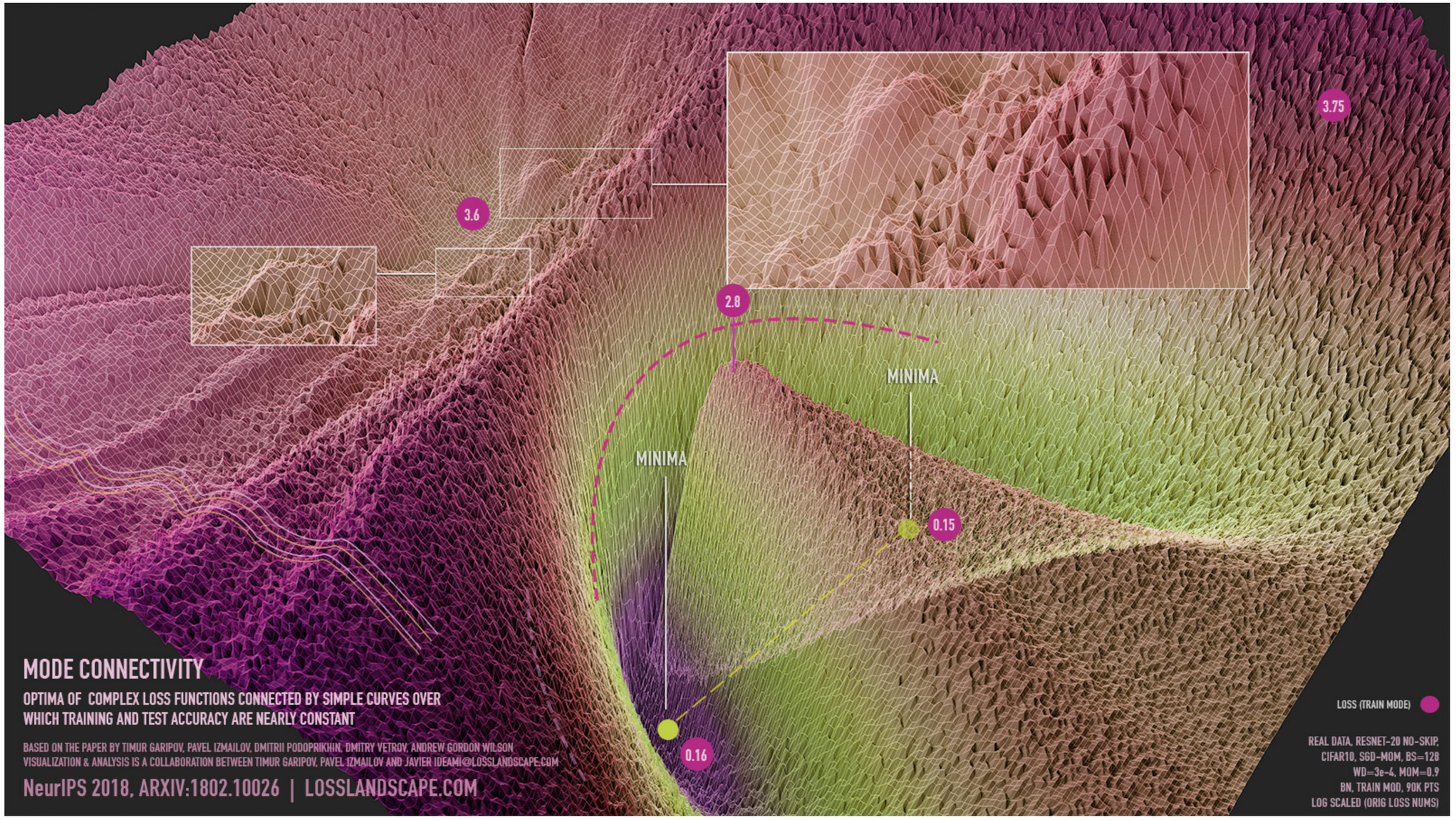


Train loss of ResNet-164 on CIFAR-100

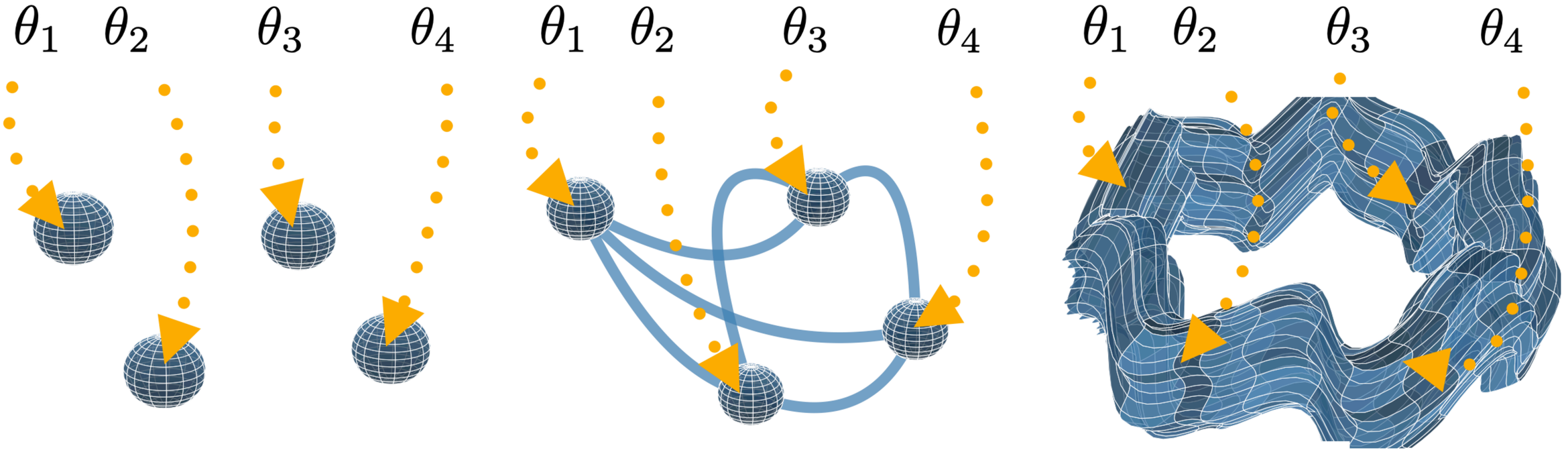
Garipov et al., 2018

Mode connectivity

https://losslandscape.com

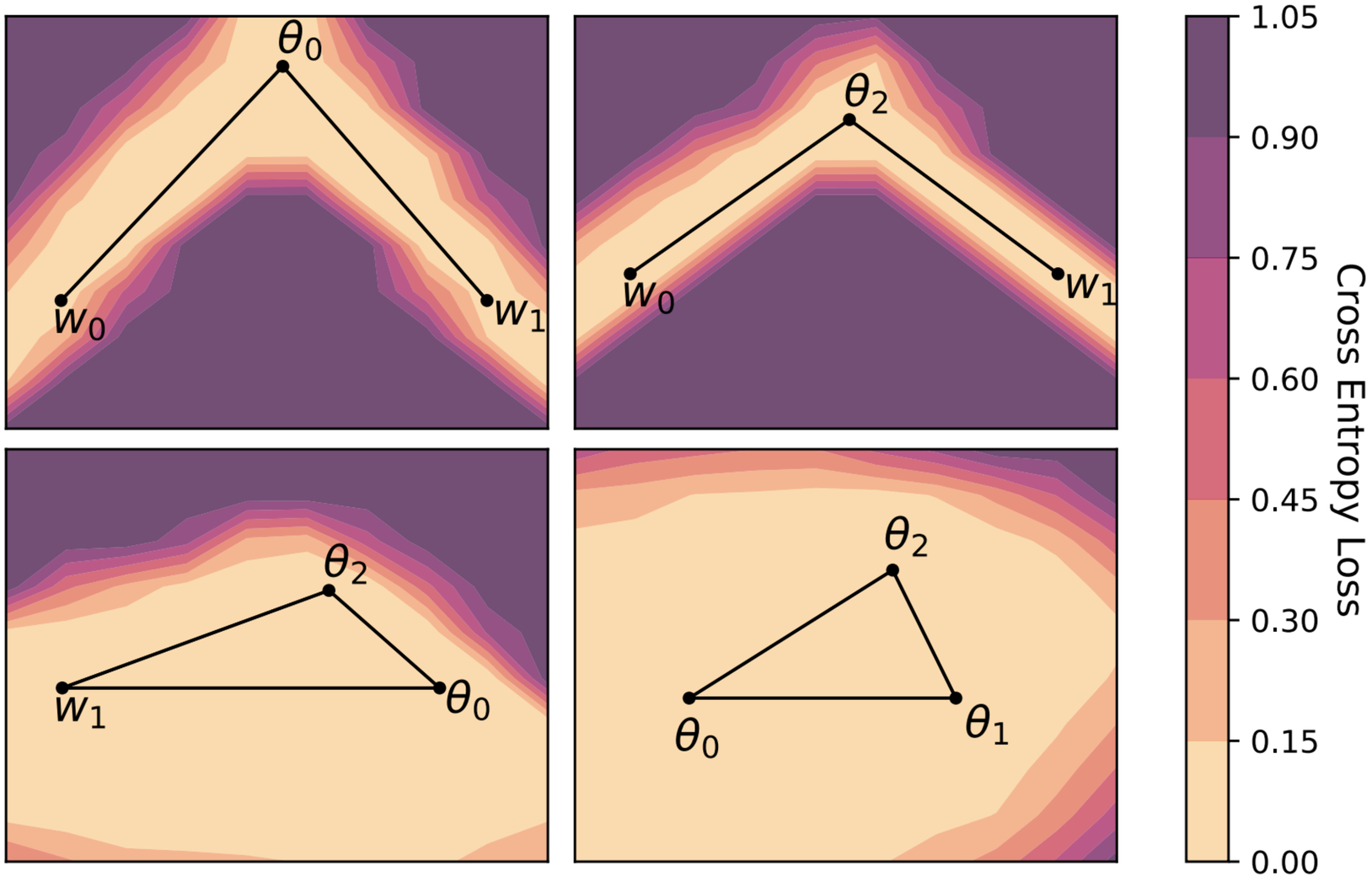


Mode connecting volumes



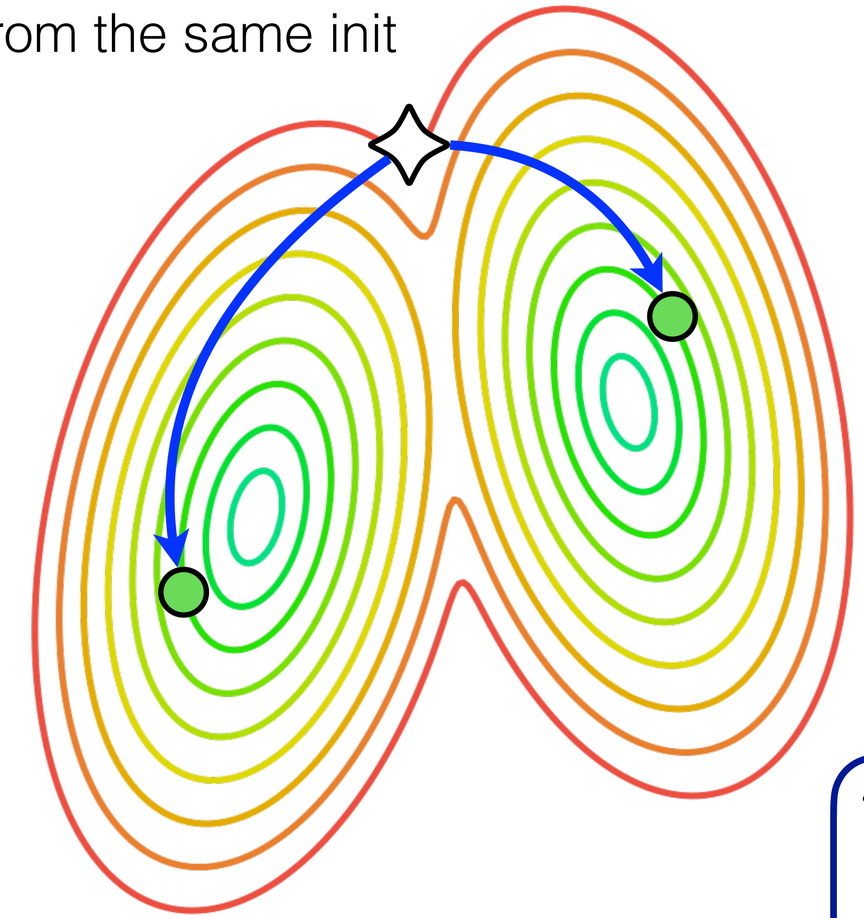
Benton et al., 2021

Mode connecting volumes

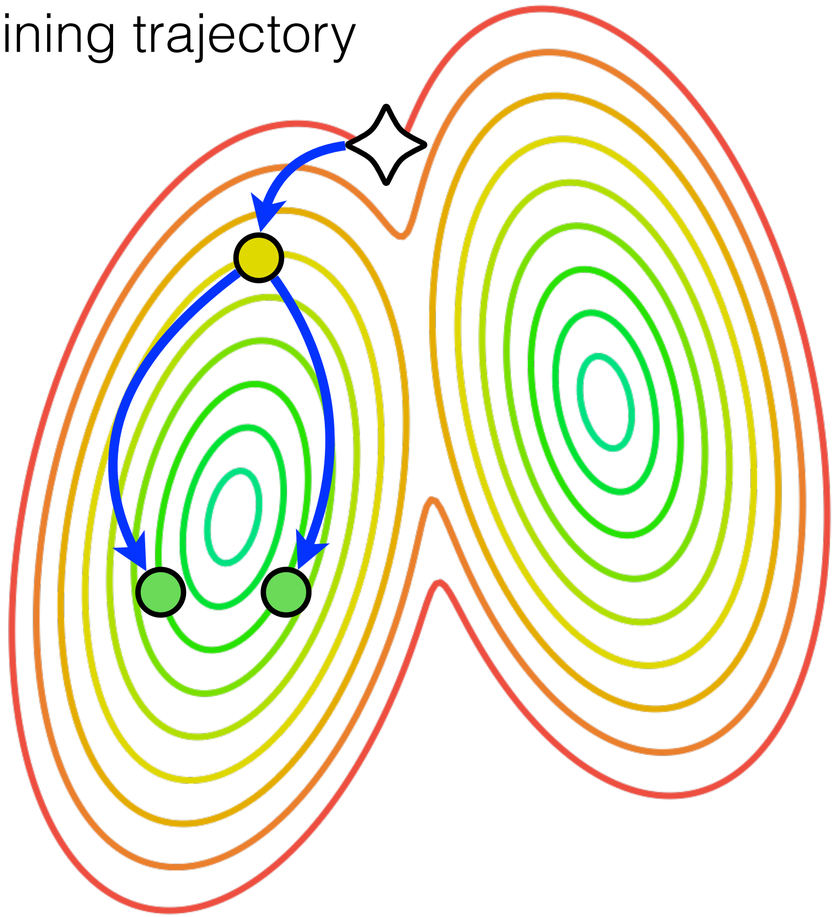


Linear Mode Connectivity (LMC)

Models trained
from the same init



Models sharing a part
of training trajectory

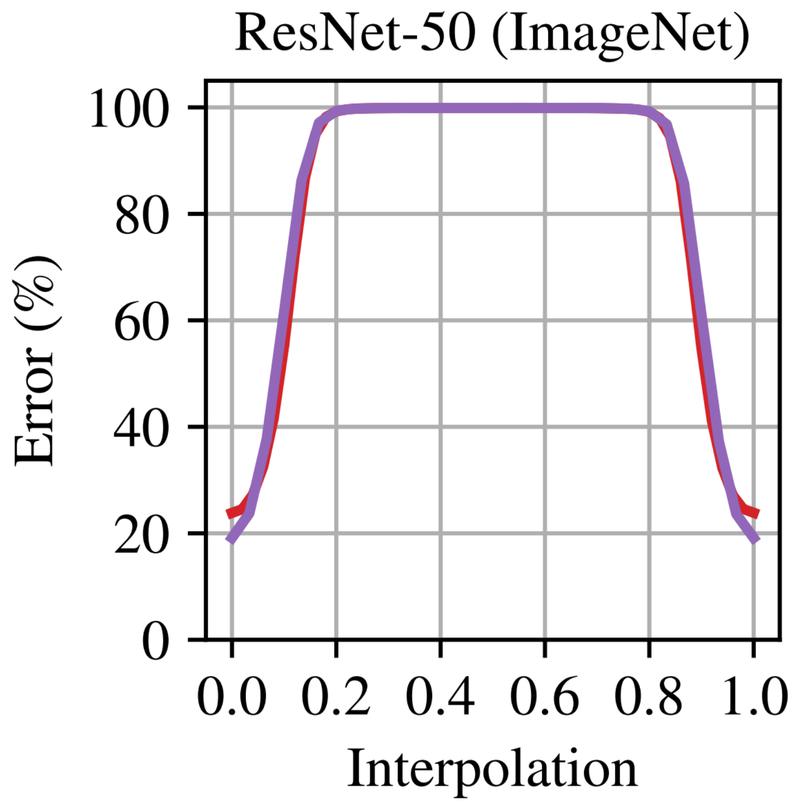


random initialization
trained model
optimization trajectory

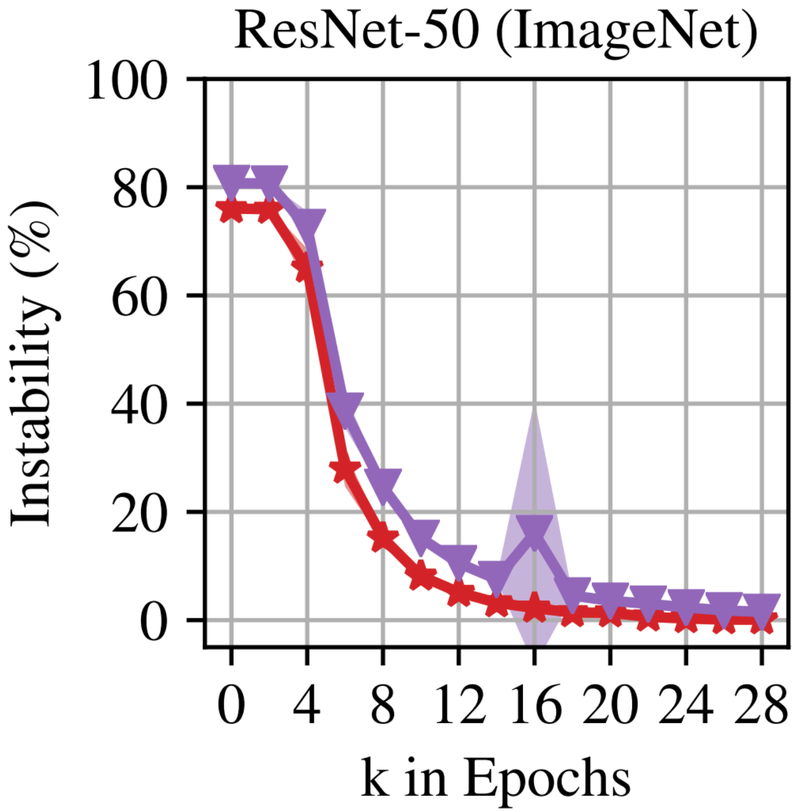
Frankle et al., 2020

Linear Mode Connectivity (LMC)

Interpolation of models trained from the same init



Error barrier between models

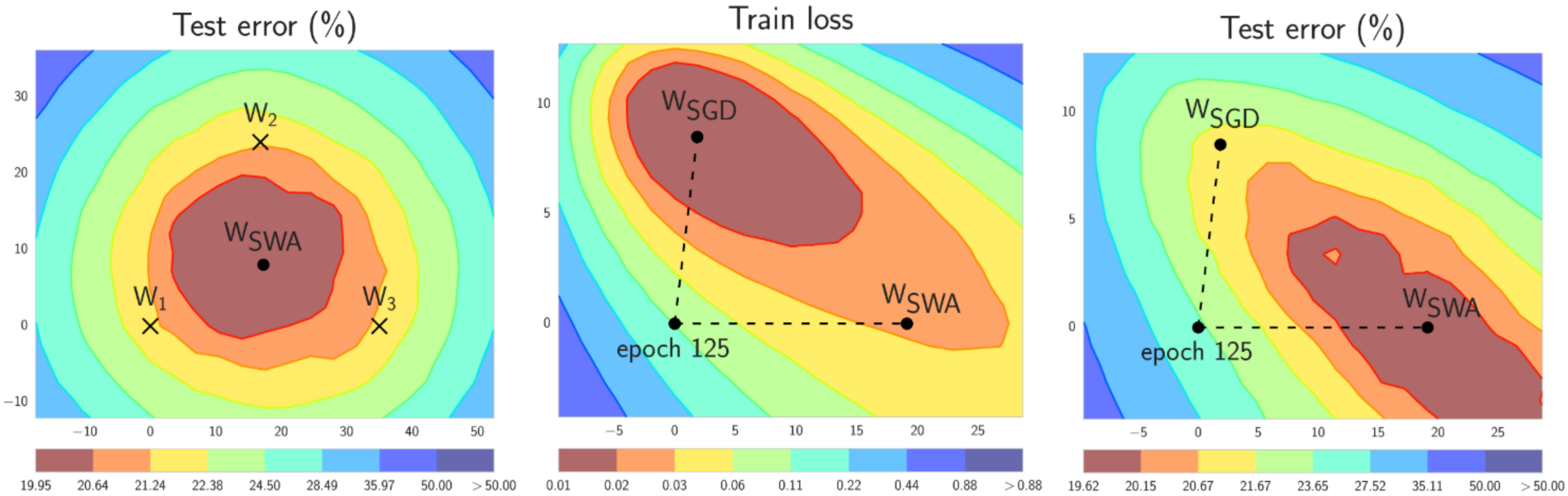


Test Error Train Error

Plan

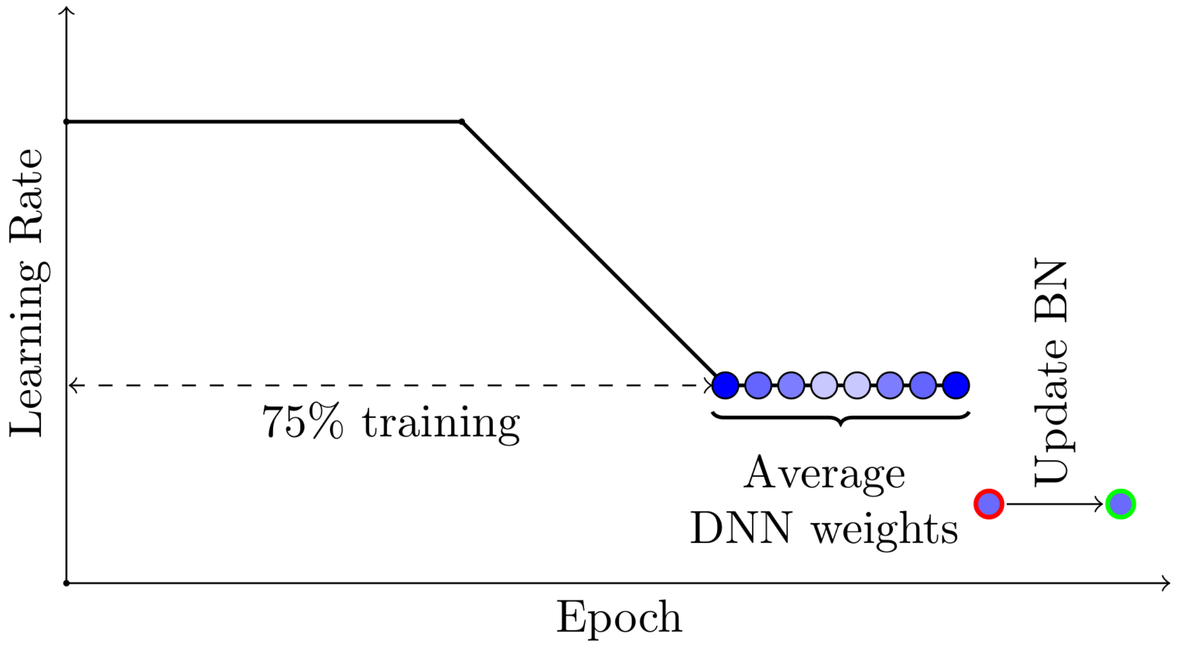
- Mode connectivity and Linear Mode Connectivity (LMC)
- **Weight averaging techniques**
  - Stochastic Weight Averaging (SWA)
  - Model soups
- Ensembling methods
  - Deep Ensembles
  - Cyclical methods: SSE, FGE, cSGLD
  - Non-cyclical methods: KFAC-Laplace, SWAG, SPRO
- Ensembles in transfer learning
- Weight-averaging based optimizers

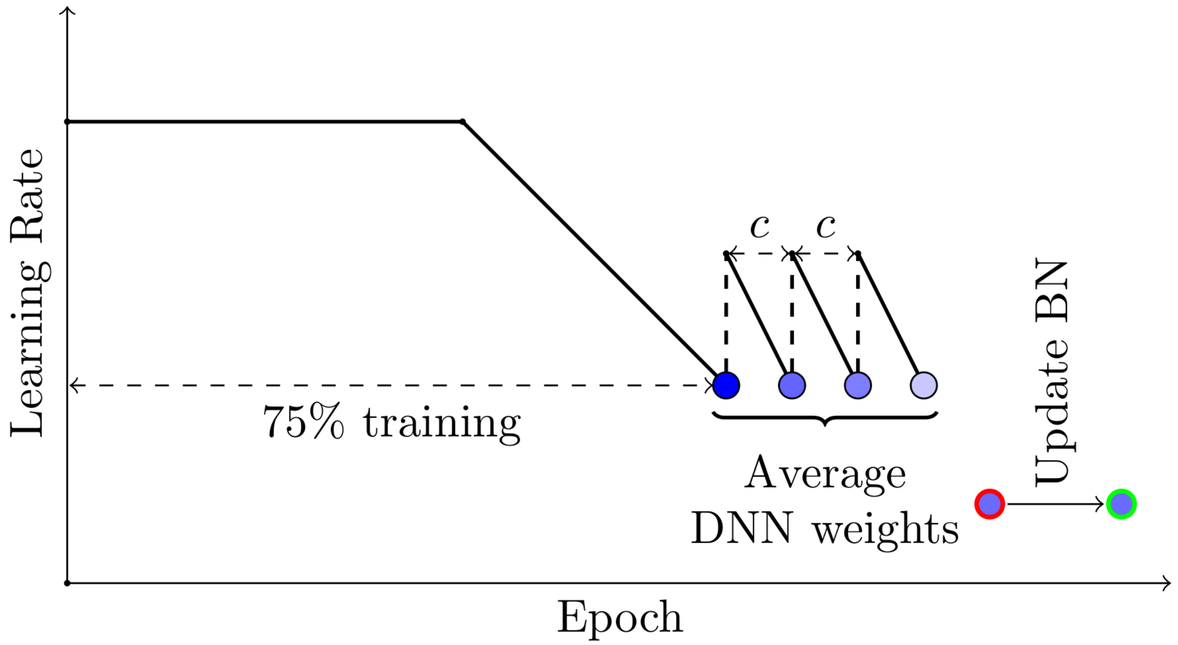
Stochastic Weight Averaging (SWA)



Izmailov et al., 2018

Stochastic Weight Averaging (SWA)





LMC in transfer learning

DomainNet, real domain

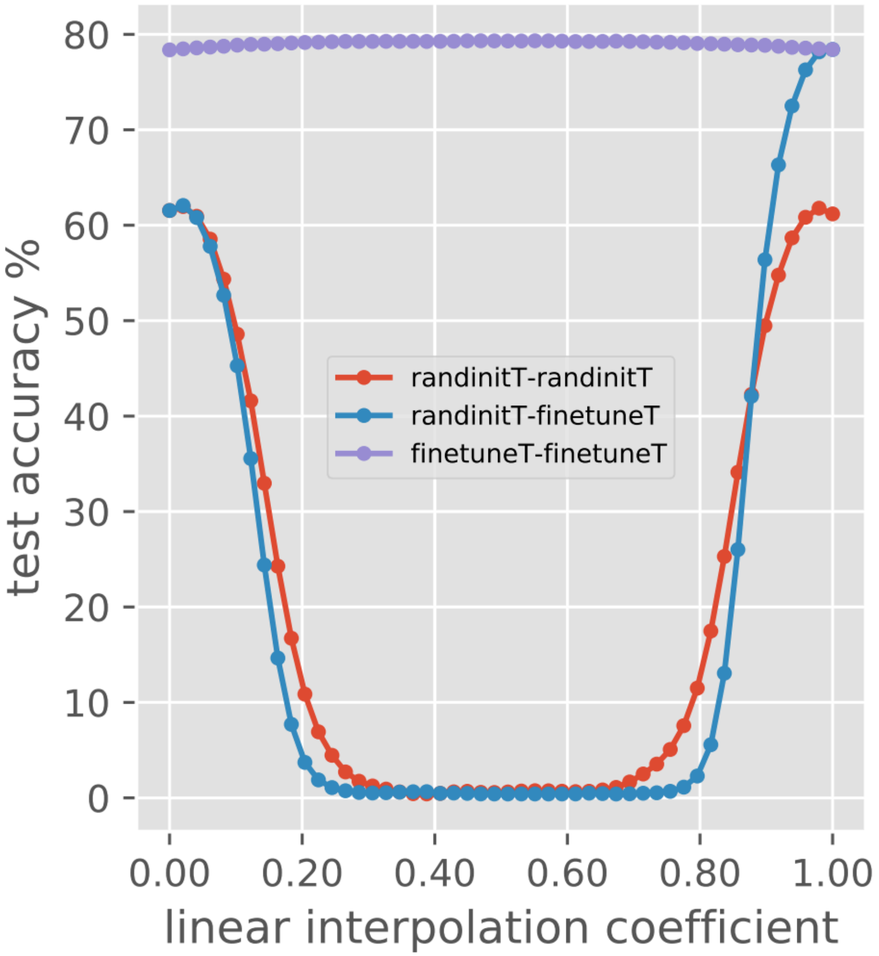


Table 1: Feature similarity for different layers of ResNet-50, target domain CHEXPERT

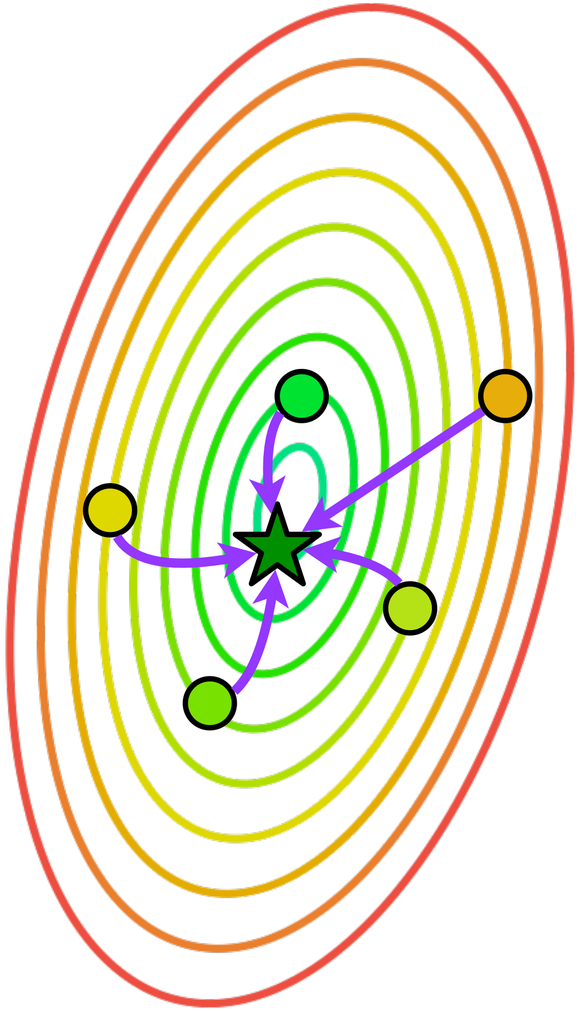
models/layer	conv1	layer 1	layer 2	layer 3	layer 4
P-T & P	0.6225	0.4592	0.2896	0.1877	0.0453
P-T & P-T	0.6710	0.8230	0.6052	0.4089	0.1628
P-T & RI-T	0.0036	0.0011	0.0022	0.0003	0.0808
RI-T & RI-T	0.0016	0.0088	0.0004	0.0004	0.0424

Table 2: Features ℓ_2 distance between two P-T and two RI-T for different target domains

domain/model	2 P-T	2 RI-T	P-T & P	RI-T & P
CHEXPERT	200.12	255.34	237.31	598.19
clipart	178.07	822.43	157.19	811.87
quickdraw	218.52	776.76	195.44	785.22
real	193.45	815.08	164.83	796.80

Neyshabur et al., 2020

Model soups

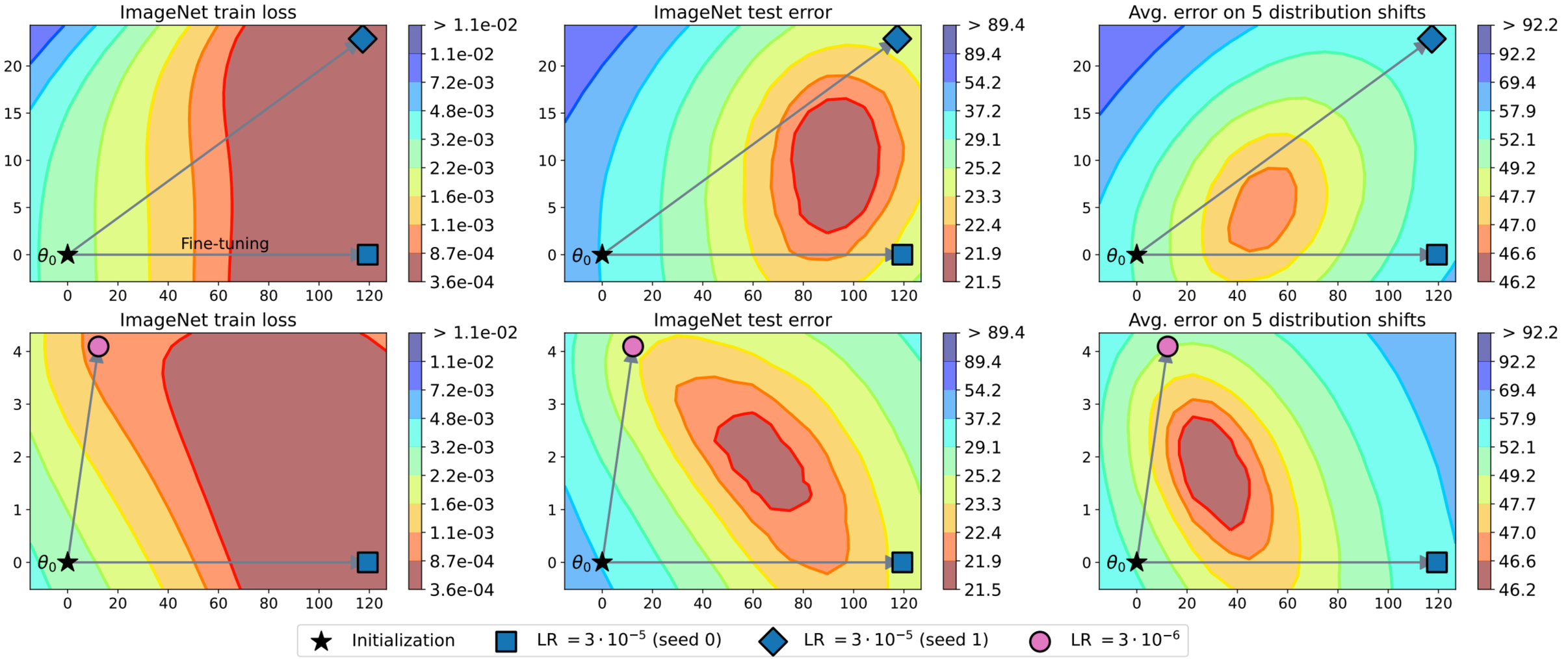


- Fine-tune a model several times with various augmentations / hyperparameters
- Average the weight of all models **(uniform soup)** or choose them greedily **(greedy soup)**

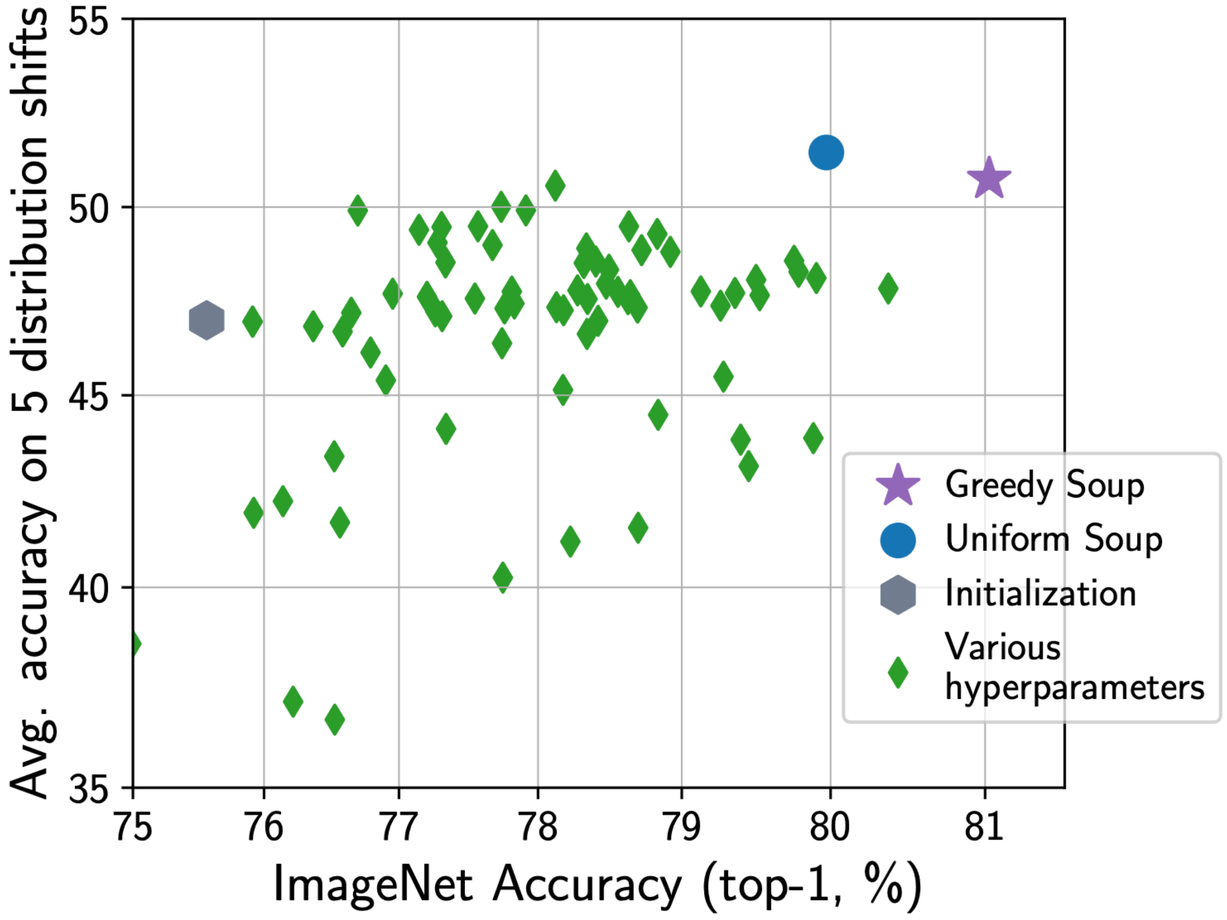
	Method	Cost
Best on val. set	$f(x, \arg\max_i \text{ValAcc}(\theta_i))$	$\mathcal{O}(1)$
Ensemble	$\frac{1}{k}\sum_{i=1}^{k} f(x, \theta_i)$	$\mathcal{O}(k)$
Uniform soup	$f\left(x, \frac{1}{k}\sum_{i=1}^{k} \theta_i\right)$	$\mathcal{O}(1)$

Wortsman et al., 2022

Model soups


ImageNet train loss
ImageNet test error
Avg. error on 5 distribution shifts
Fine-tuning
θ_0
> 1.1e-02
1.1e-02
7.2e-03
4.8e-03
3.2e-03
2.2e-03
1.6e-03
1.1e-03
8.7e-04
3.6e-04
> 89.4
89.4
54.2
37.2
29.1
25.2
23.3
22.4
21.9
21.5
> 92.2
92.2
69.4
57.9
52.1
49.2
47.7
47.0
46.6
46.2
Initialization
LR $= 3 \cdot 10^{-5}$ (seed 0)
LR $= 3 \cdot 10^{-5}$ (seed 1)
LR $= 3 \cdot 10^{-6}$

Model soups



	ImageNet	Dist. shifts
Best individual model	80.38	47.83
Second best model	79.89	43.87
Uniform soup	79.97	51.45
Greedy soup	81.03	50.75
Greedy soup (random order)	80.79 (0.05)	51.30 (0.16)
Learned soup	80.89	51.07
Learned soup (by layer)	81.37	50.87
Ensemble	81.19	50.77
Greedy ensemble	81.90	49.44

Plan

- Mode connectivity and Linear Mode Connectivity (LMC)
- Weight averaging techniques
  - Stochastic Weight Averaging (SWA)
  - Model soups
- **Ensembling methods**
  - Deep Ensembles
  - Cyclical methods: SSE, FGE, cSGLD
  - Non-cyclical methods: KFAC-Laplace, SWAG, SPRO
- Ensembles in transfer learning
- Weight-averaging based optimizers

Deep Ensembles

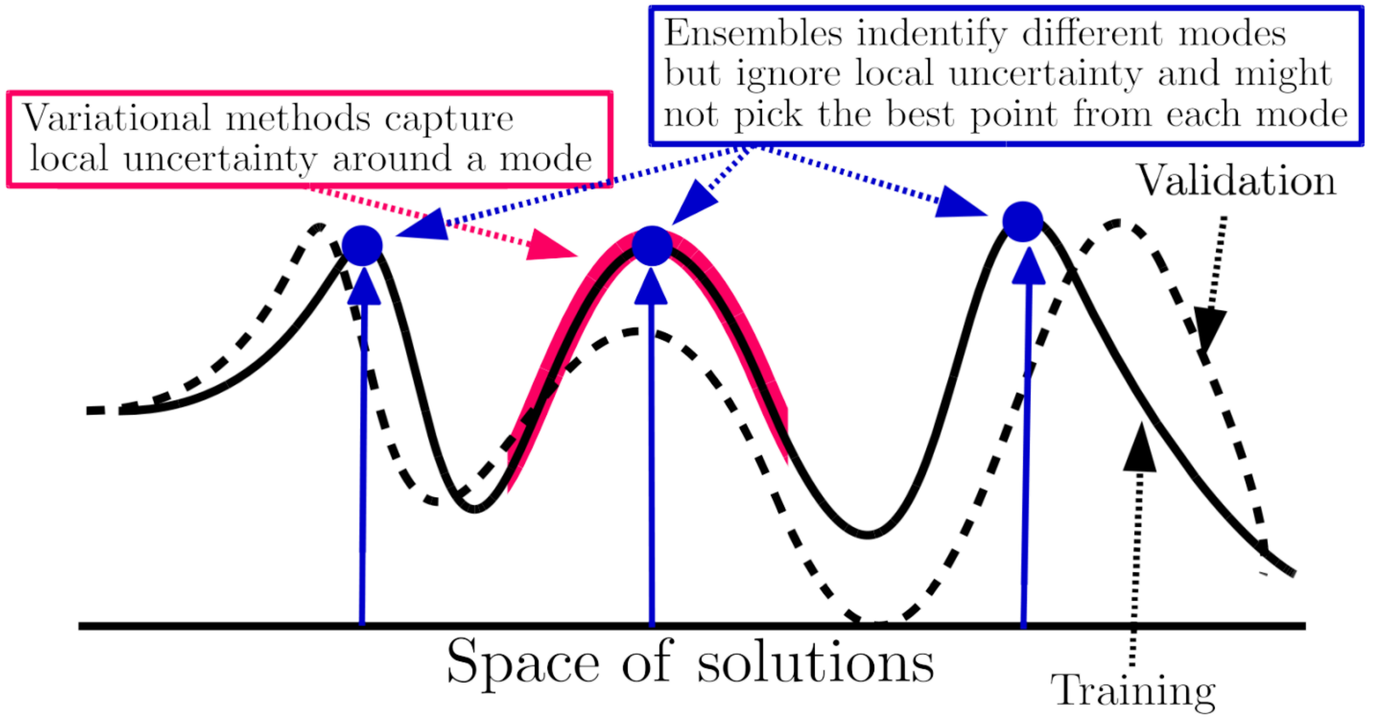


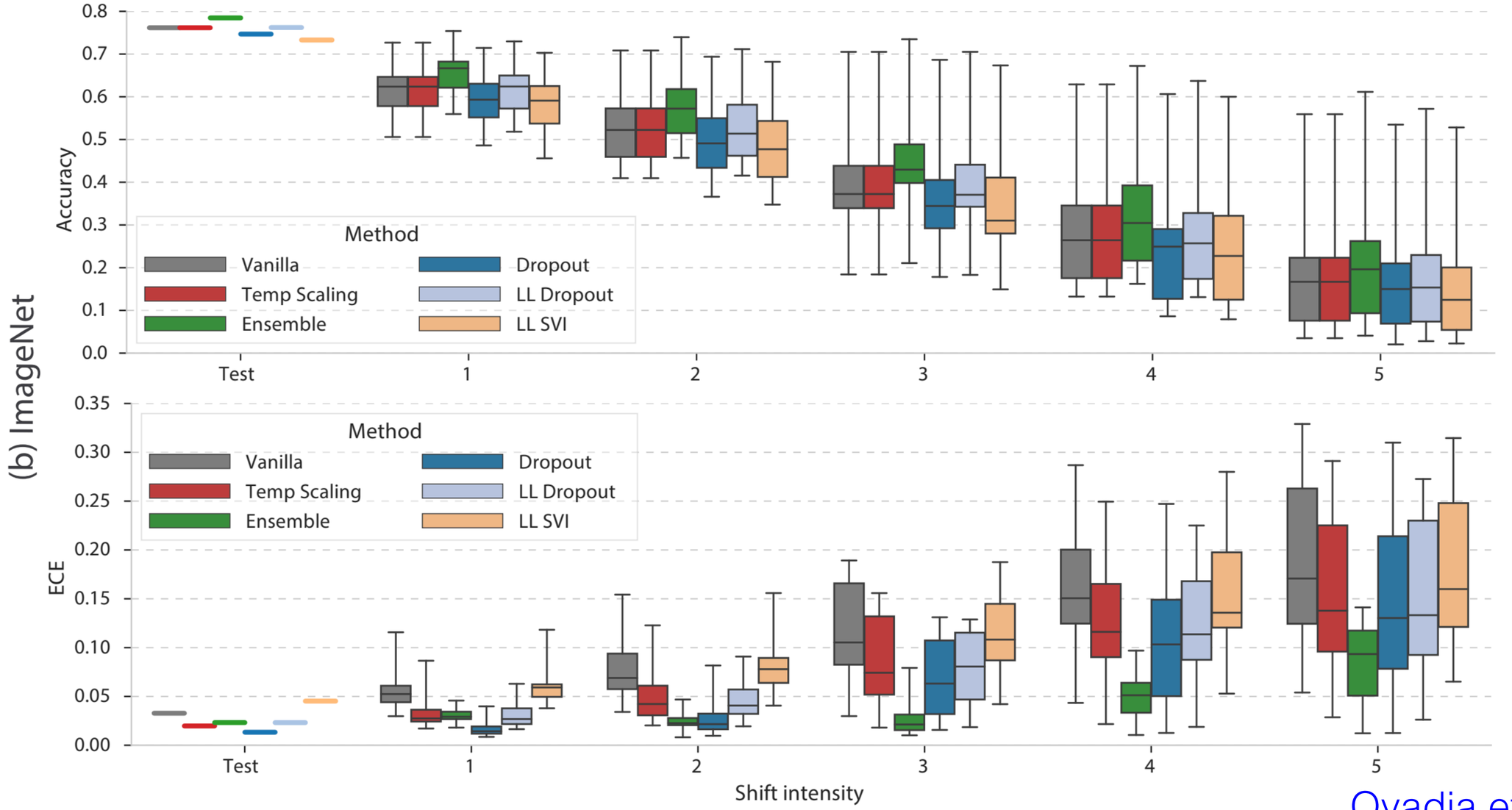
Figure 1: Cartoon illustration of the hypothesis. x-axis indicates parameter values and y-axis plots the negative loss $-L(\boldsymbol{\theta}, \{\boldsymbol{x}_n, y_n\}_{n=1}^N)$ on train and validation data.

- Train several models from different random inits
- No need to do train on bootstrapped samples
- Different models explore different modes of the loss landscape

Lakshminarayanan et al., 2017 Fort et al., 2019

Deep Ensembles

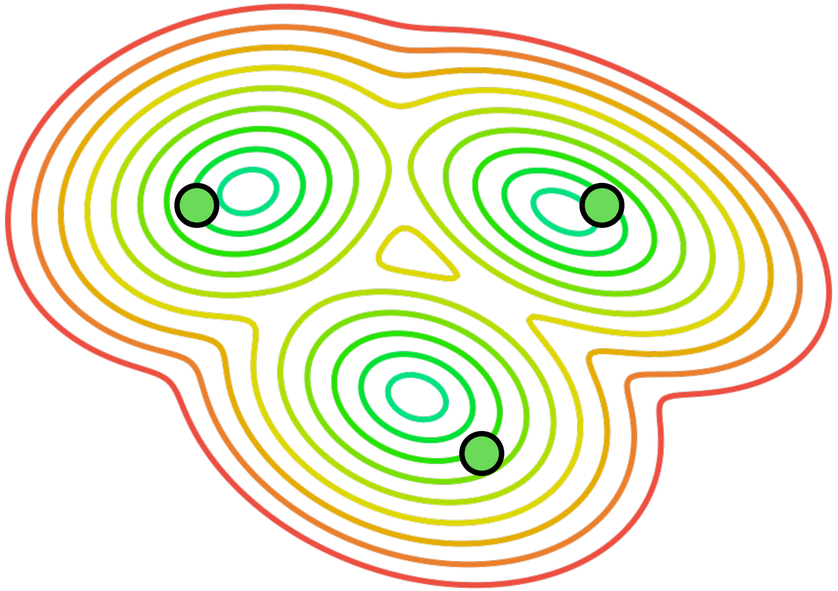
$$\mathrm{ECE} = \sum_{s=1}^{S} \frac{|B_s|}{N} |\,\mathrm{acc}(B_s) - \mathrm{conf}(B_s)|$$



Ovadia et al., 2019

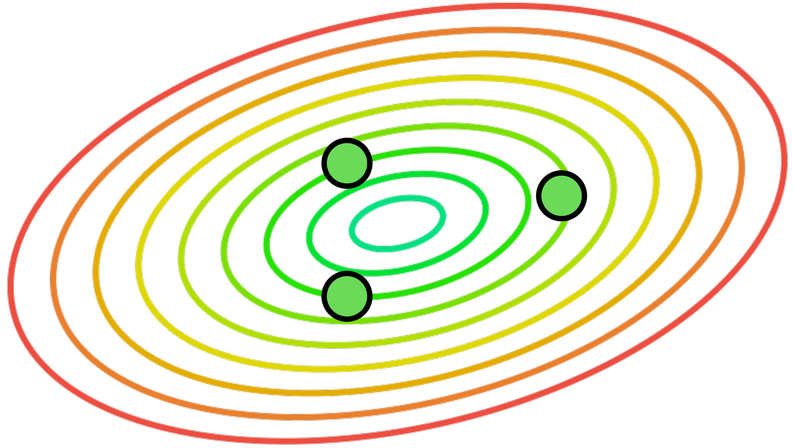
Ensembling methods

Global



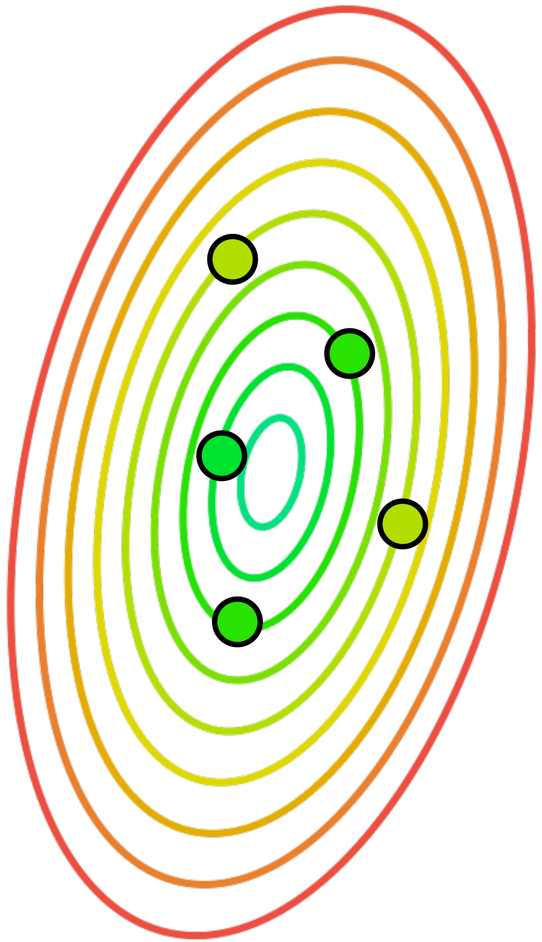
- Cover several modes of the loss landscape

Local



- Sample from a single low-loss basin

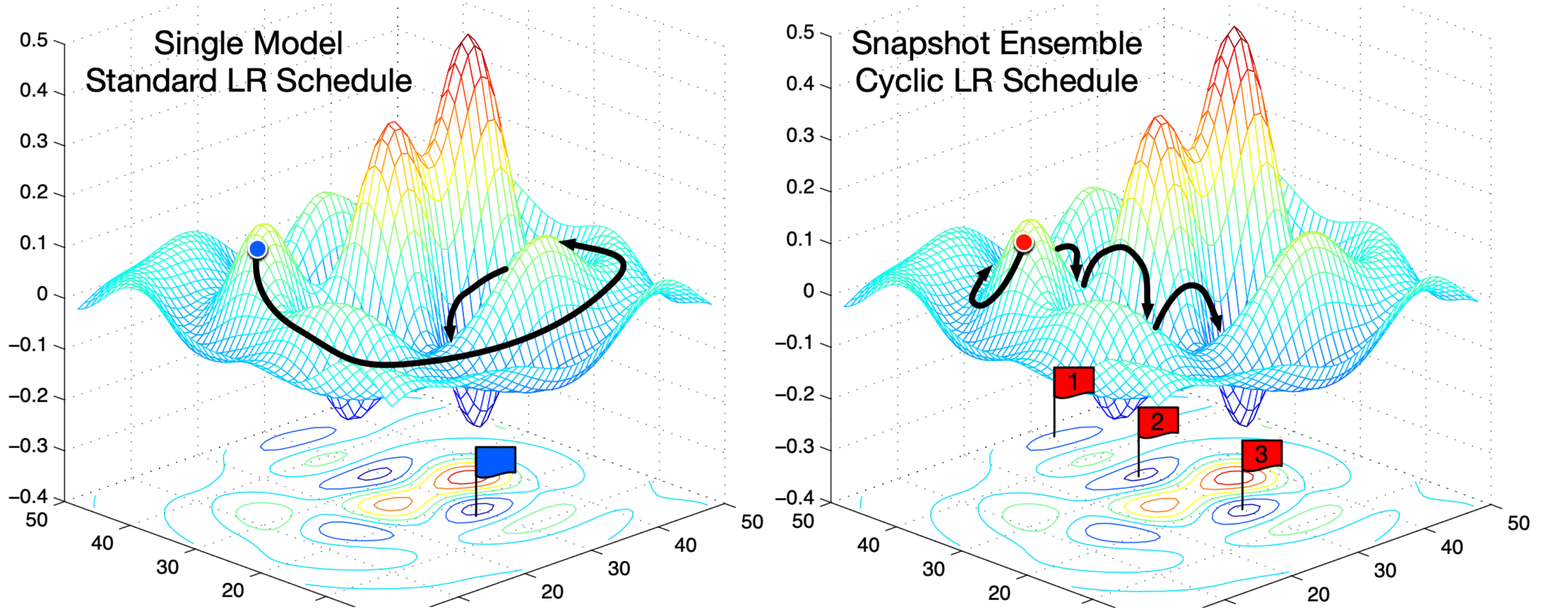
Effective ensembles in non-transfer setup

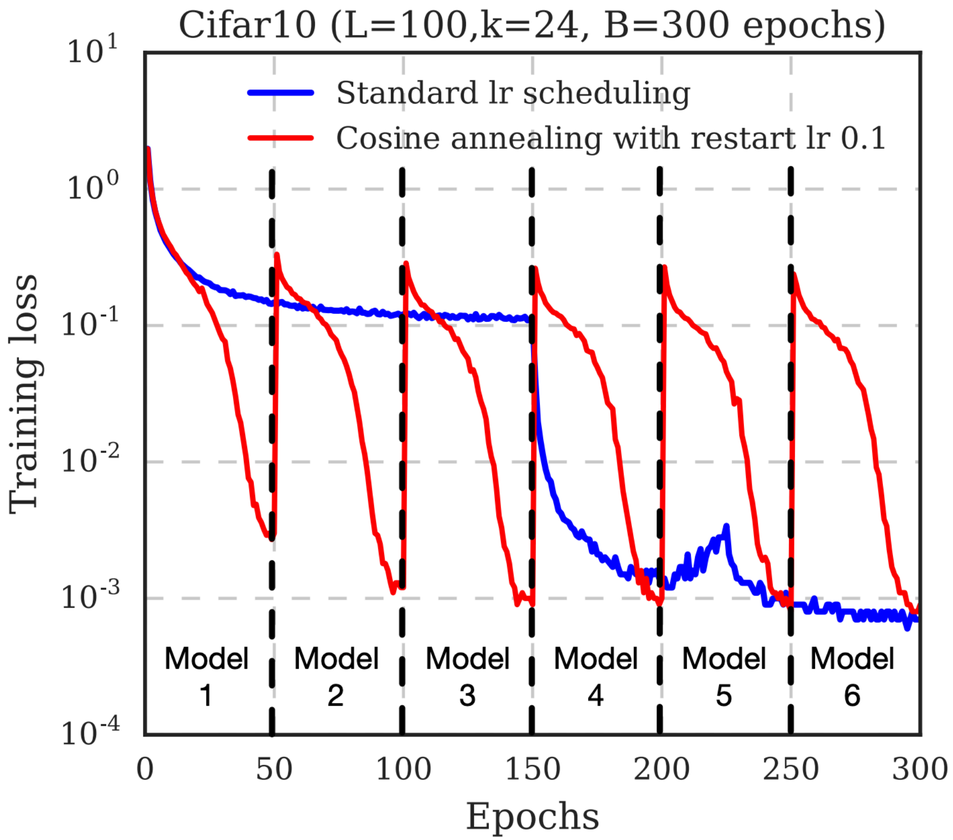


Possible approaches:

- Approximate the basin with some distribution and sample from it:
  - KFAC Laplace
  - SWA-Gaussian
  - SPRO (simplexes)
- Explore the basin using cyclical LR:
  - SSE
  - FGE
  - cSGLD

SnapShot Ensembles (SSE)





- Cyclical learning rate (LR) schedule, ensemble checkpoint at LR minima

Huang et al., 2017

SnapShot Ensembles (SSE)

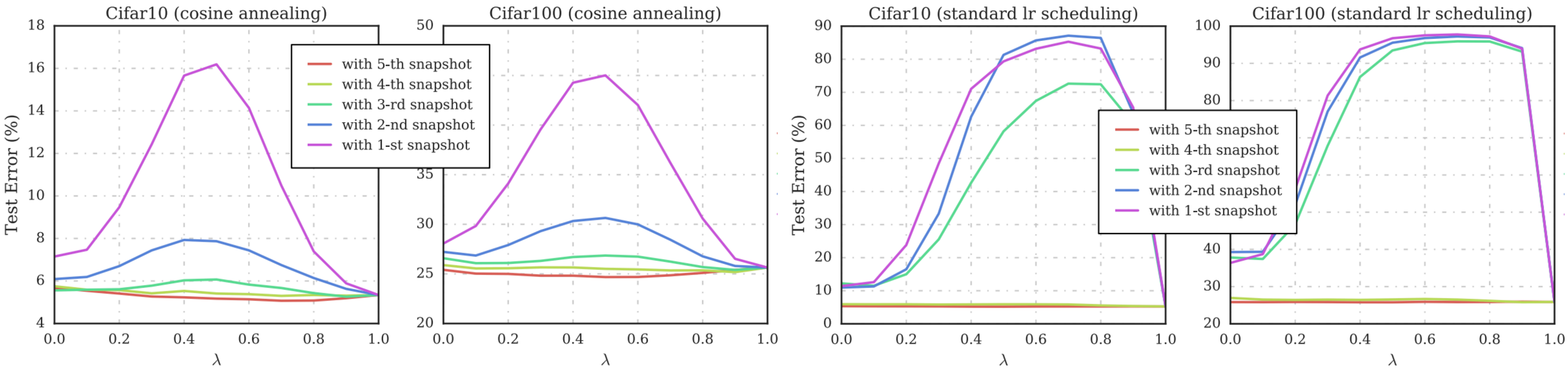
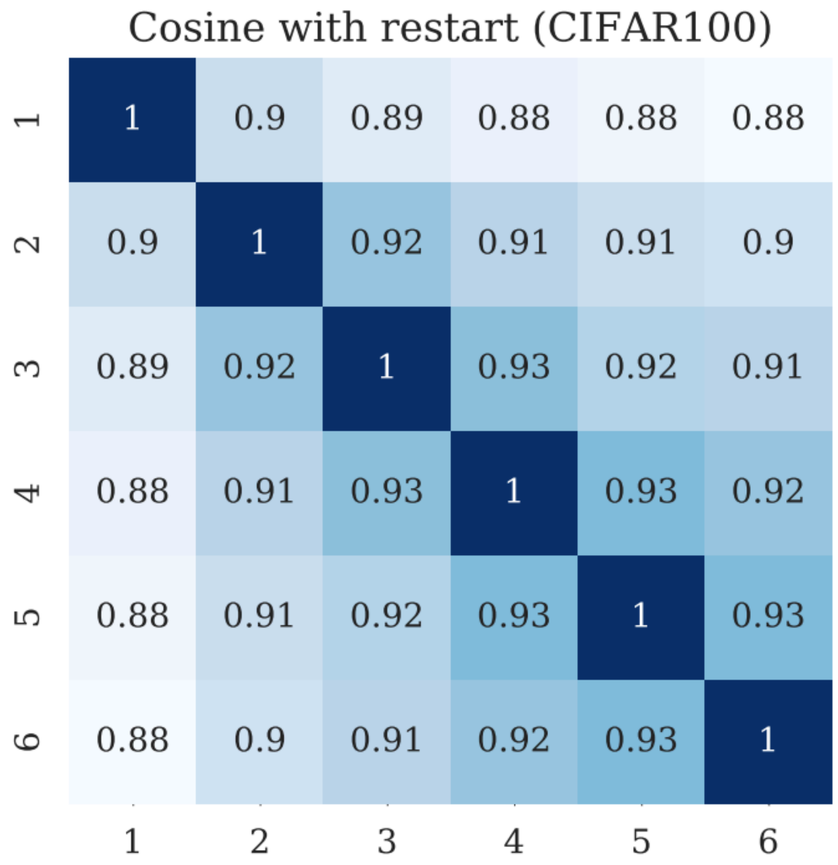
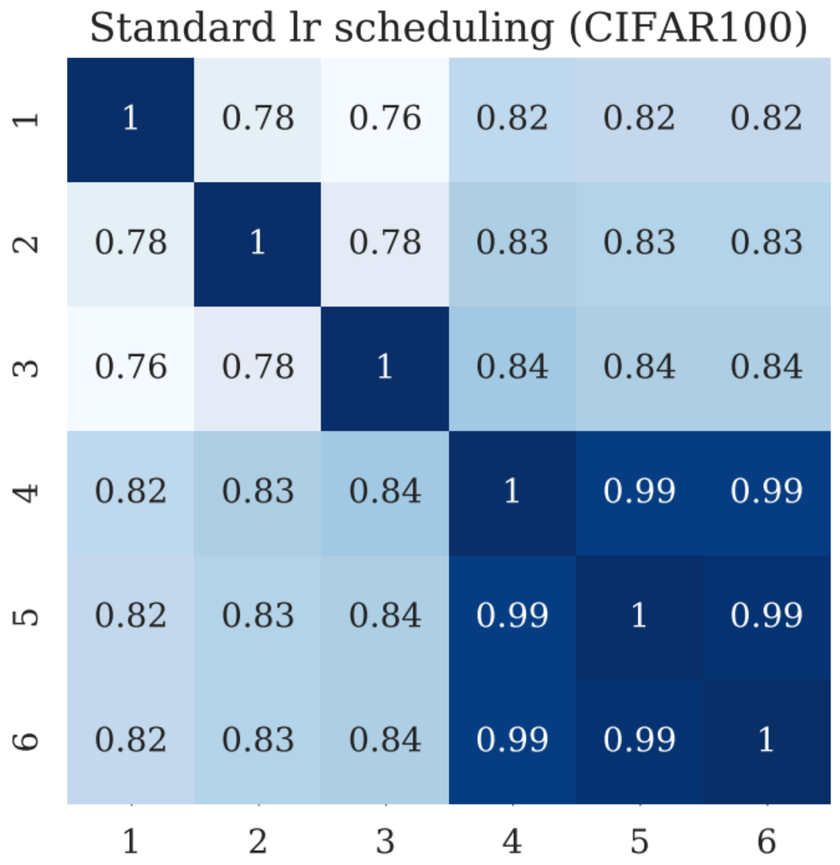


Figure 5: Interpolations in parameter space between the final model (sixth snapshot) and all intermediate snapshots. $\lambda = 0$ represents an intermediate snapshot model, while $\lambda = 1$ represents the final model. **Left:** A Snapshot Ensemble, with cosine annealing cycles ($\alpha_0 = 0.2$ every $B/M = 50$ epochs). **Right:** A NoCycle Snapshot Ensemble, (two learning rate drops, snapshots every 50 epochs).

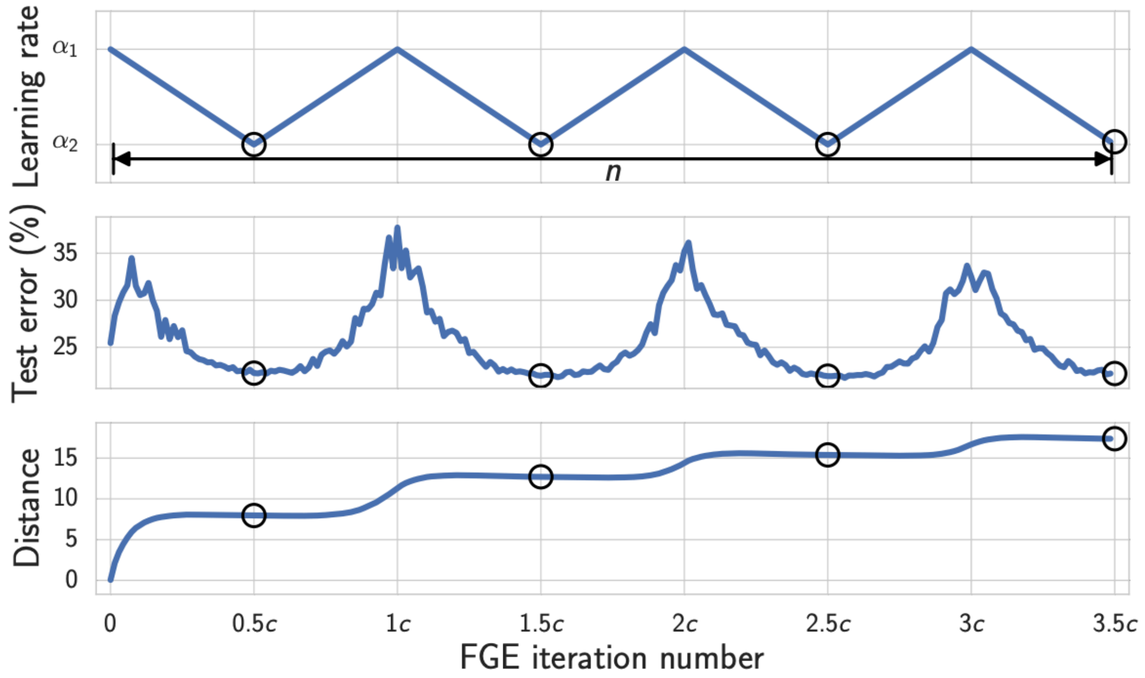
SnapShot Ensembles (SSE)

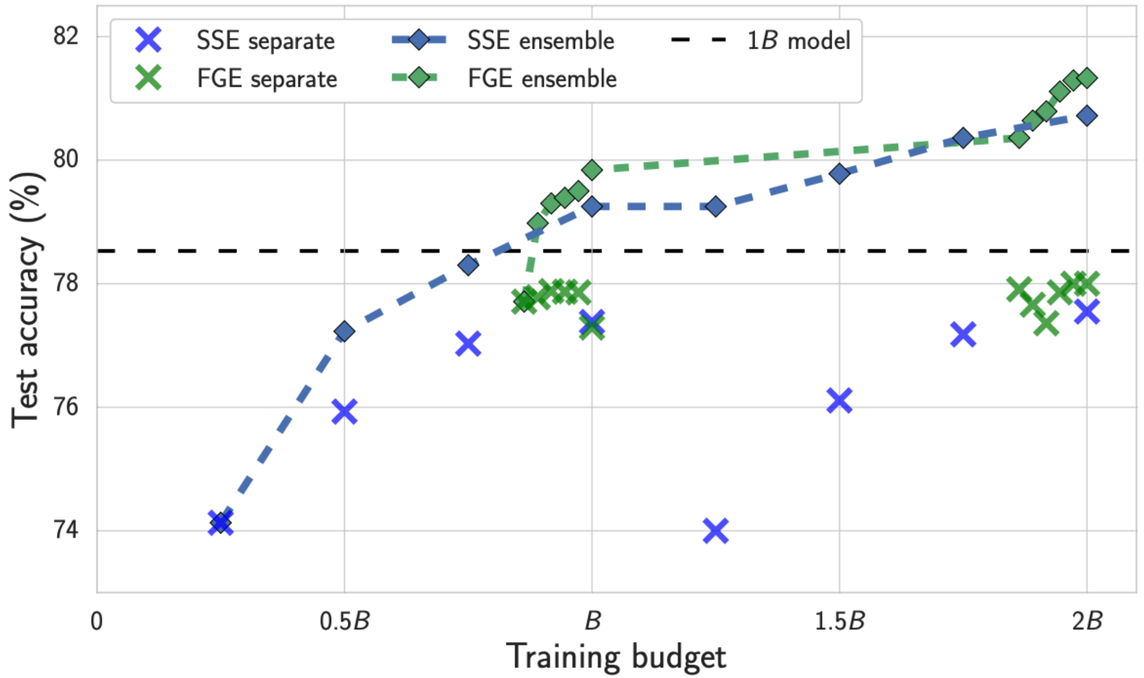




Pairwise correlation of softmax outputs
between DenseNet-100 snapshots

Fast Geometric Ensembles (FGE)

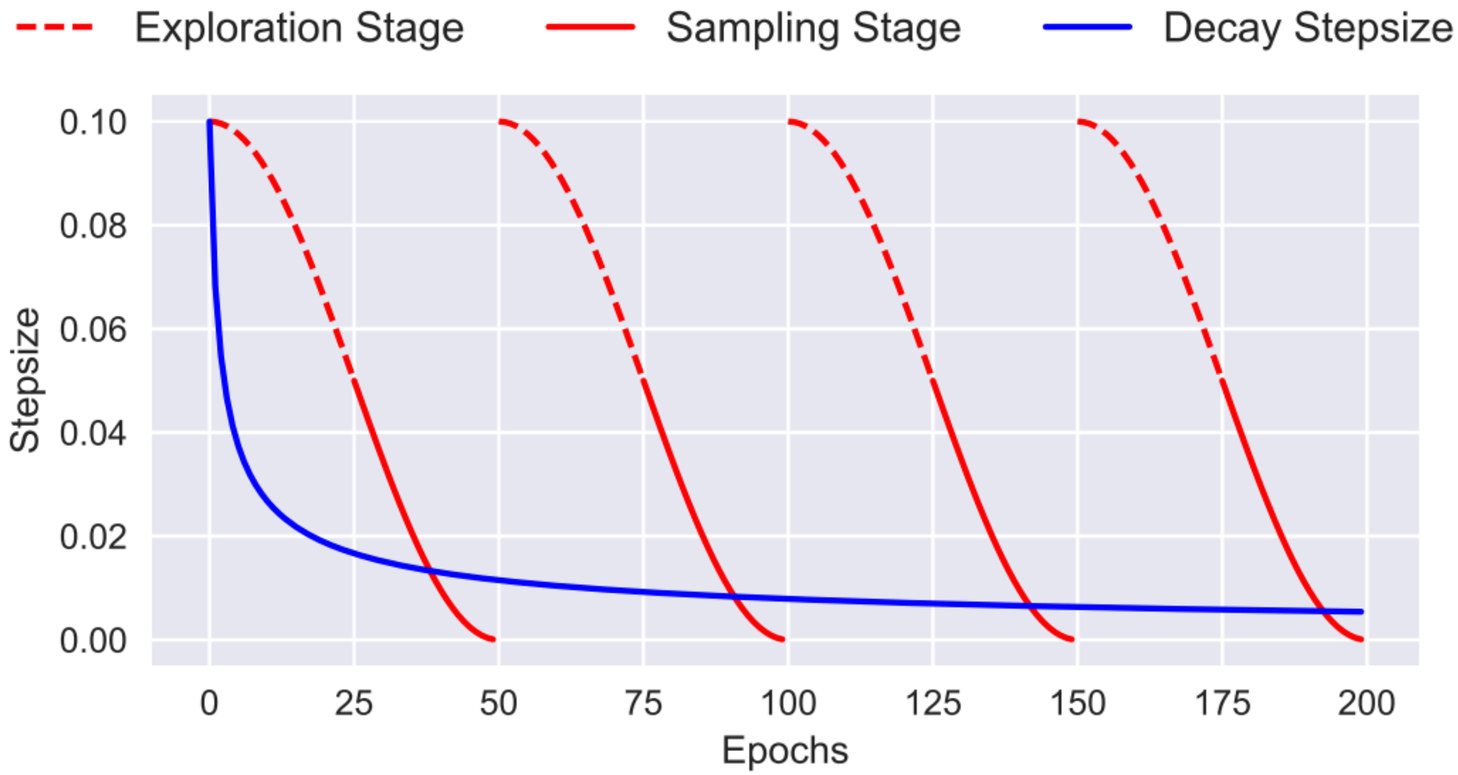


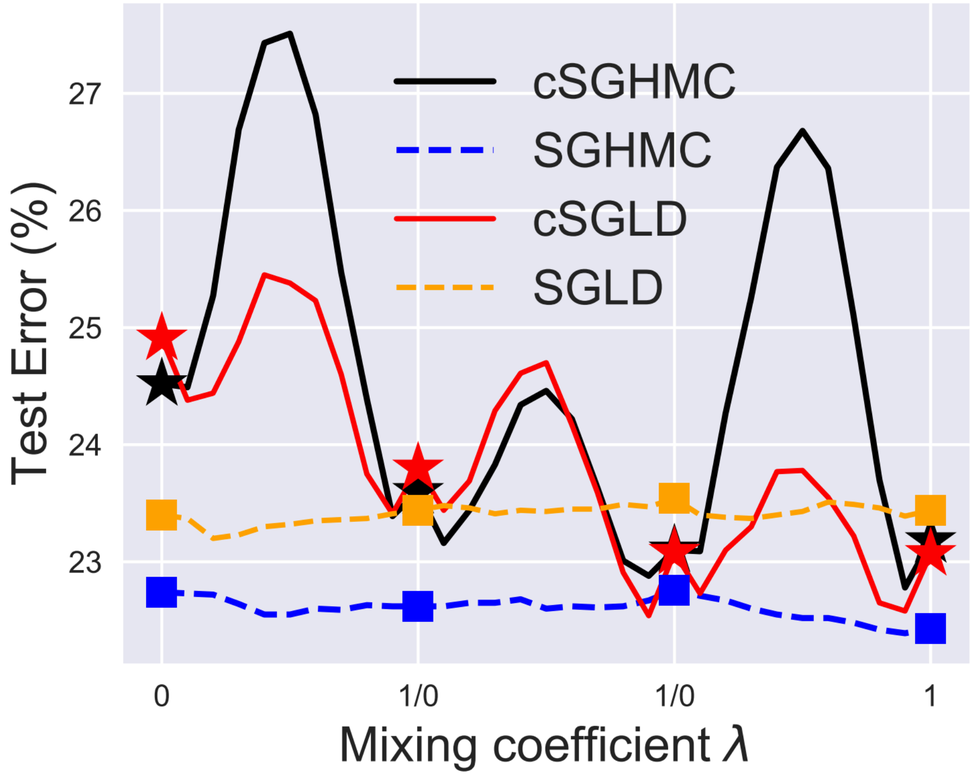


- Triangular schedule, sampling locally from a single basin

[Garipov et al., 2018]

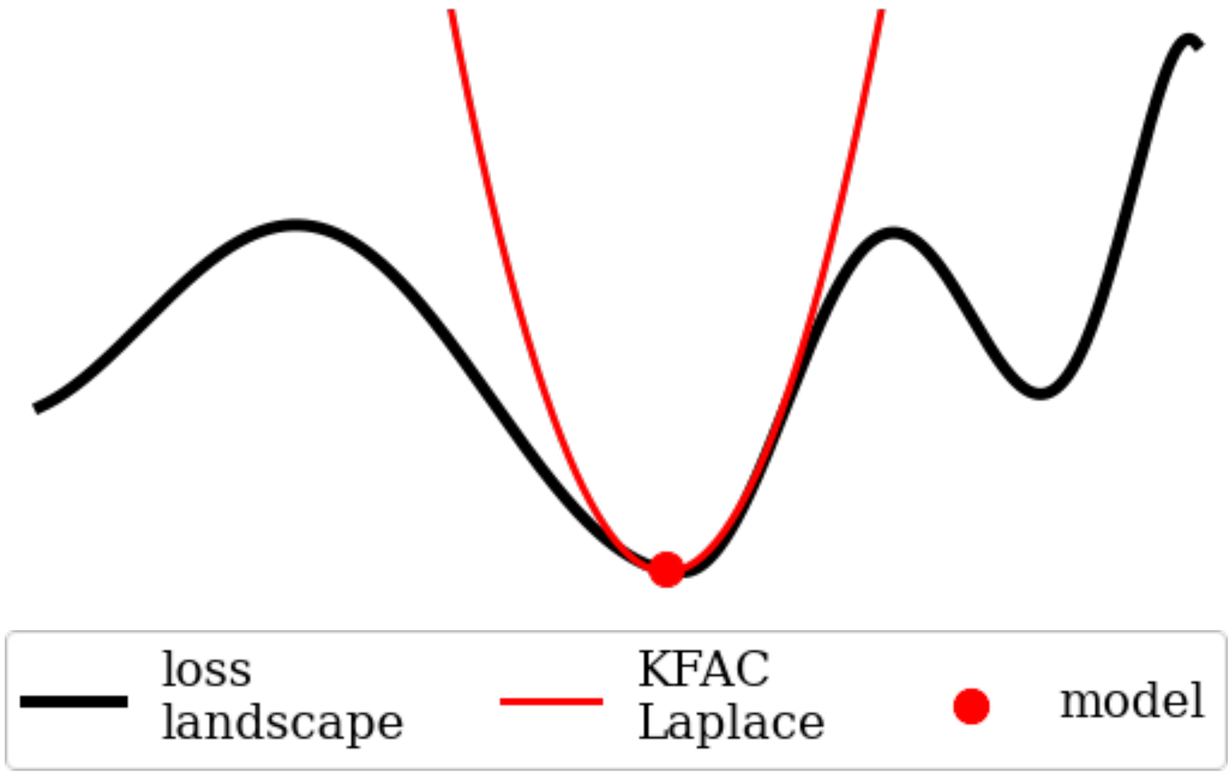
SSE + SGLD = cSGLD





[Zhang et al., 2020]

KFAC Laplace

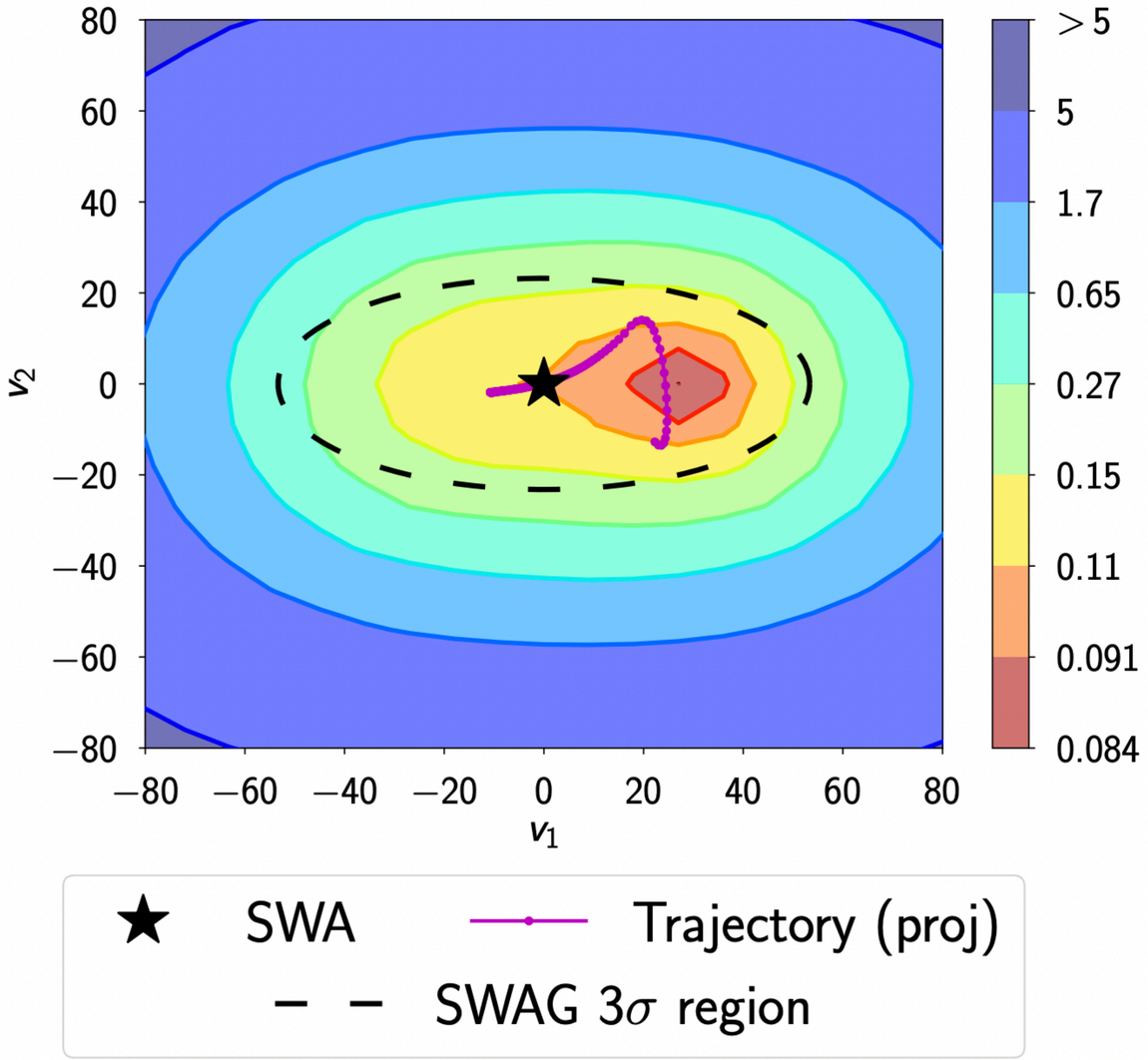


KFAC Laplace (Ritter et al, 2018)

- Fit a Gaussian around a single trained model
- Kronecker factored approximation of Hessian matrix as covariance
- Sample new ensemble models from the Gaussian

Ritter et al., 2018

SWAG

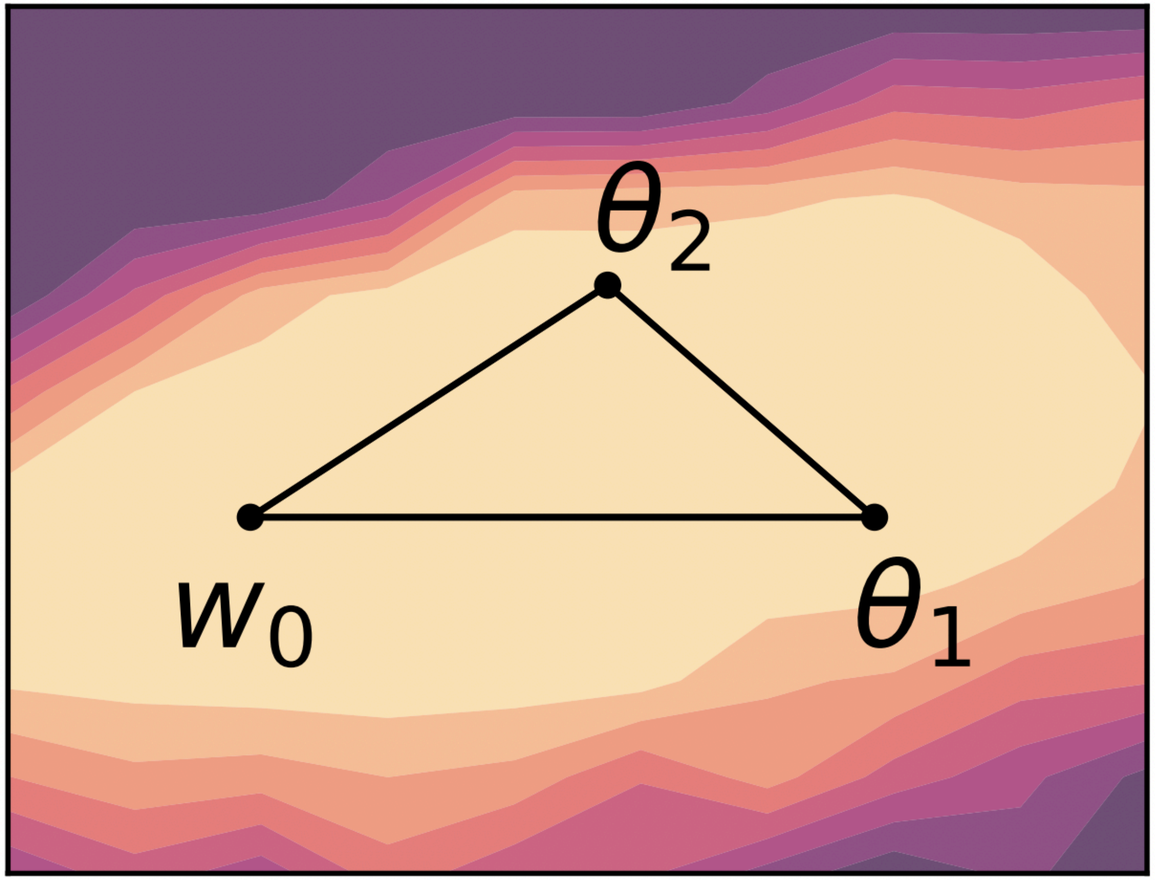


SWA-Gaussian (SWAG)

- Fit a Gaussian over models from training trajectory (SWA models)
- Requires additional epochs of training
- Sample new ensemble models from the Gaussian

Maddox et al., 2019

SPRO (Simplexes)

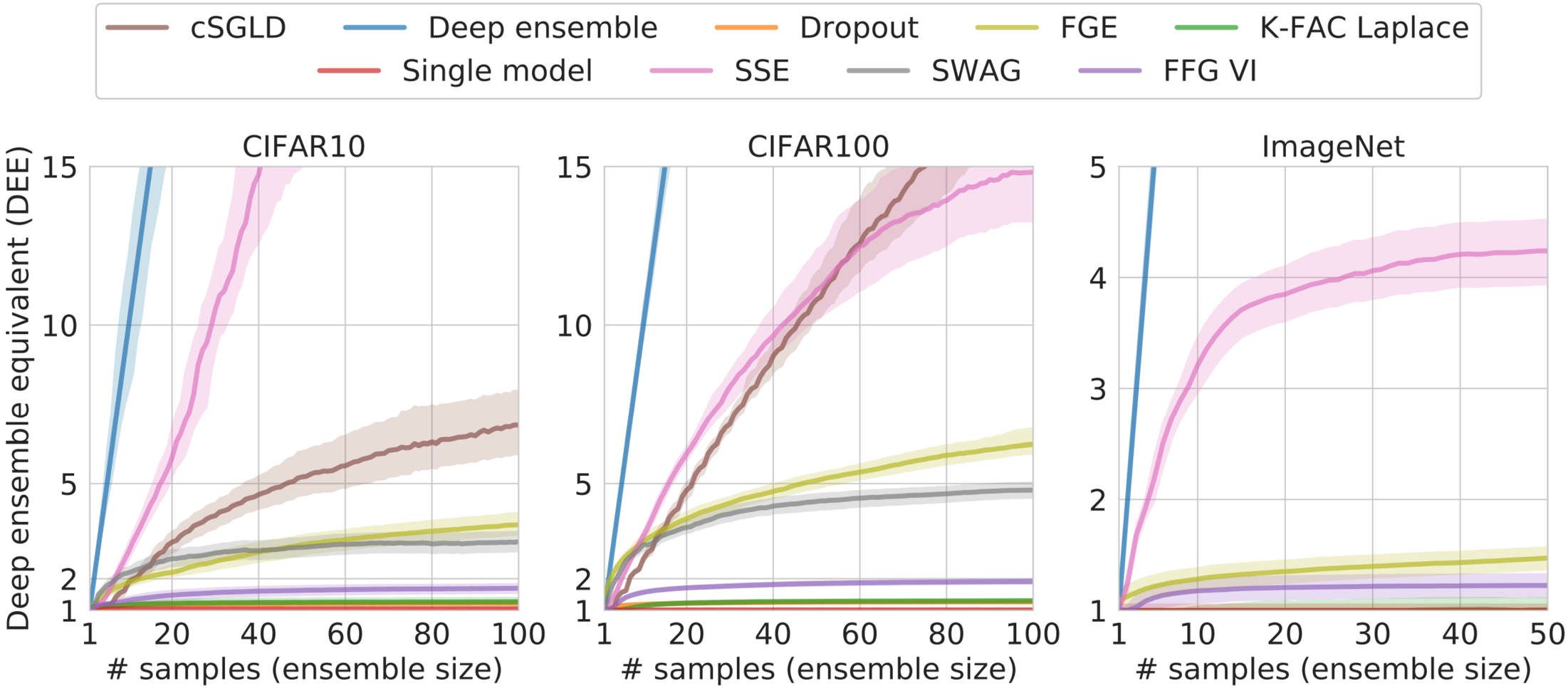


Simplicial Pointwise Random Optimization (SPRO)

- Fit a simplex (e.g., a triangle) in the vicinity of a trained model
- Requires additional epochs of training
- Sample new ensemble models from the simplex

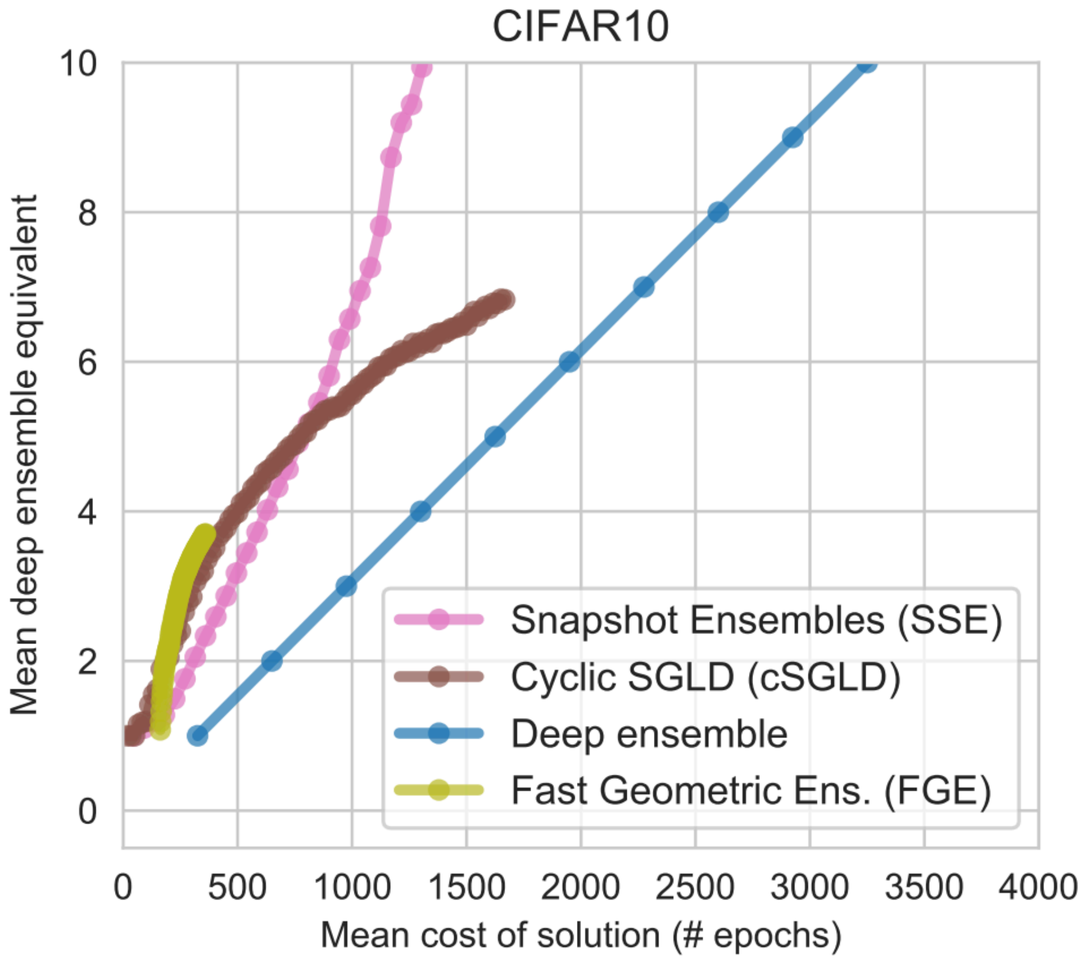
Benton et al., 2021

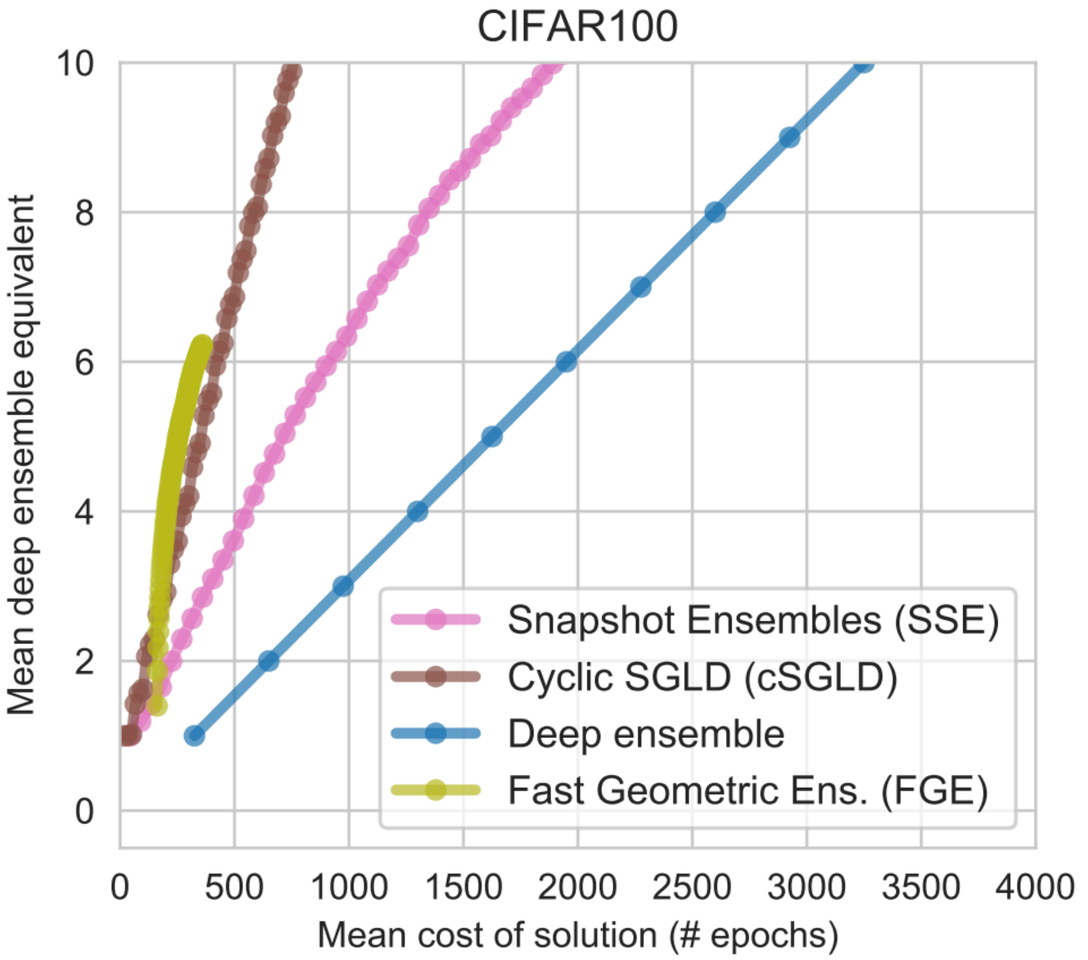
Deep Ensemble Equivalent (DEE)



[Ashukha et al., 2020]

Deep Ensemble Equivalent (DEE)

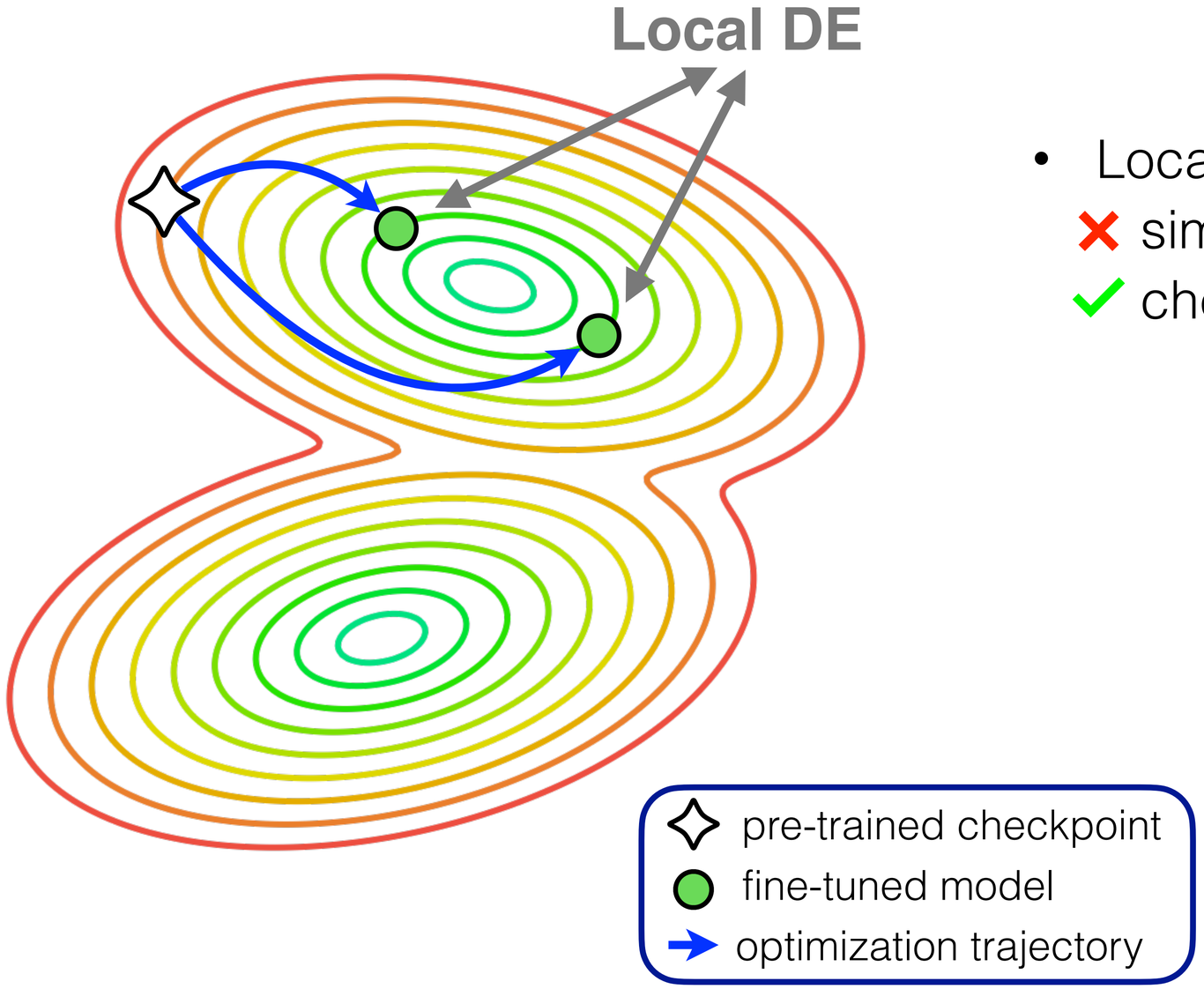




Plan

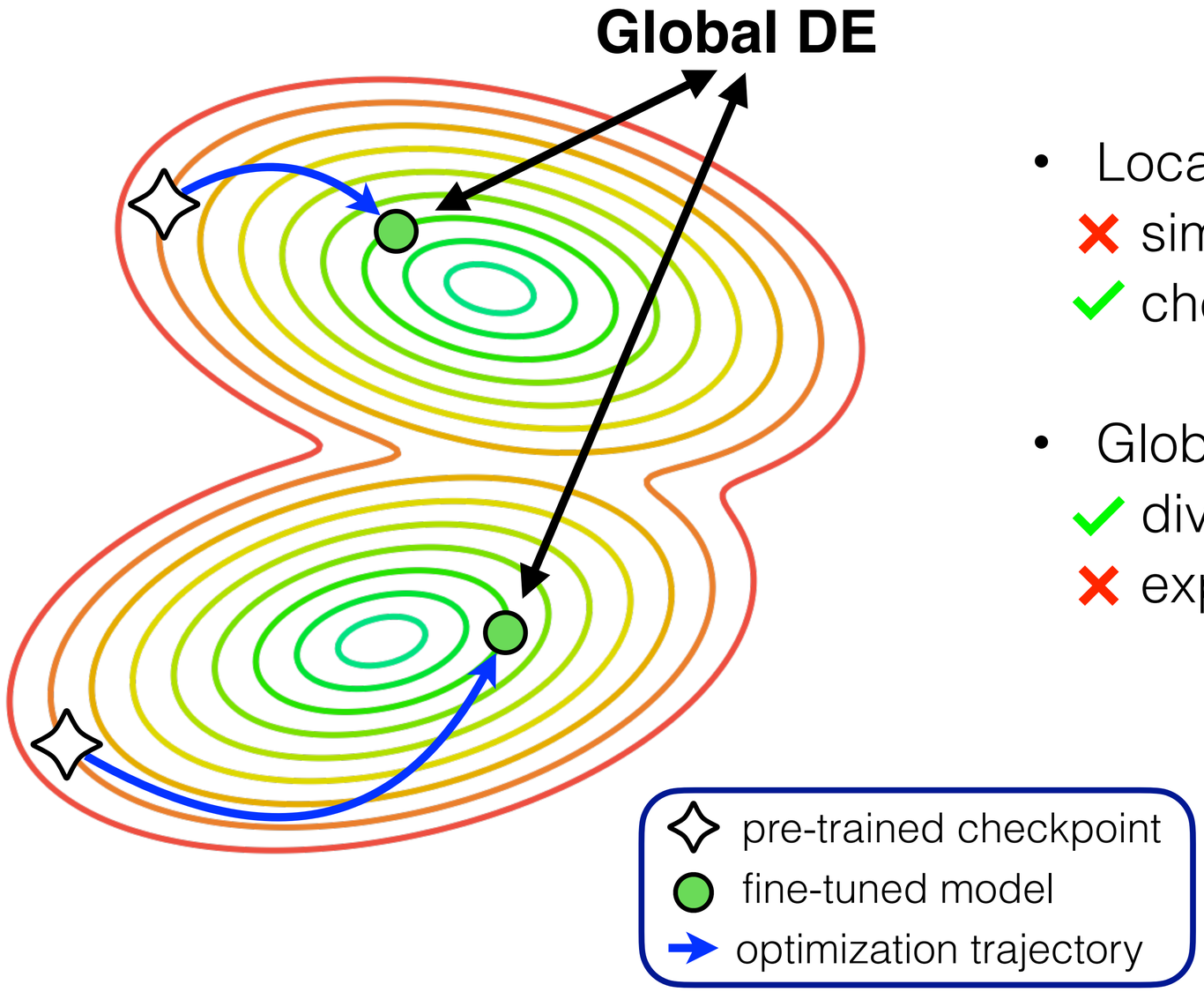
- Mode connectivity and Linear Mode Connectivity (LMC)
- Weight averaging techniques
  - Stochastic Weight Averaging (SWA)
  - Model soups
- Ensembling methods
  - Deep Ensembles
  - Cyclical methods: SSE, FGE, cSGLD
  - Non-cyclical methods: KFAC-Laplace, SWAG, SPRO
- **Ensembles in transfer learning**
- Weight-averaging based optimizers

Ensembles in transfer learning



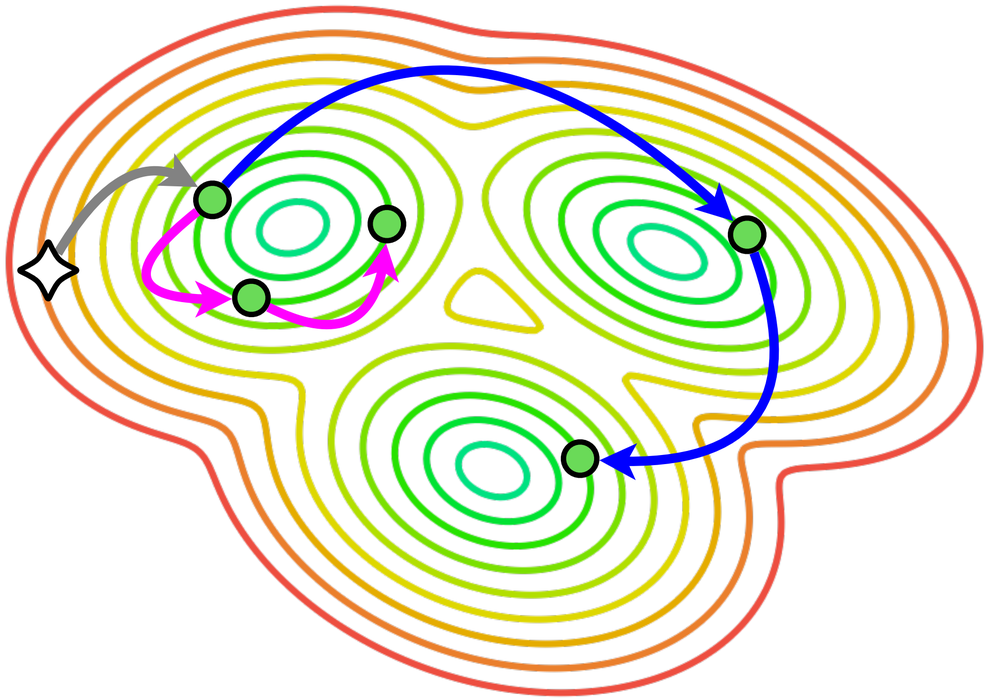
- Local Deep Ensemble (**Local DE**)
  - ✗ similar networks, lower quality
  - ✓ cheap to train

Ensembles in transfer learning



- Local Deep Ensemble (**Local DE**)
  - ✗ similar networks, lower quality
  - ✓ cheap to train
- Global Deep Ensemble (**Global DE**)
  - ✓ diverse networks, higher quality
  - ✗ expensive to train

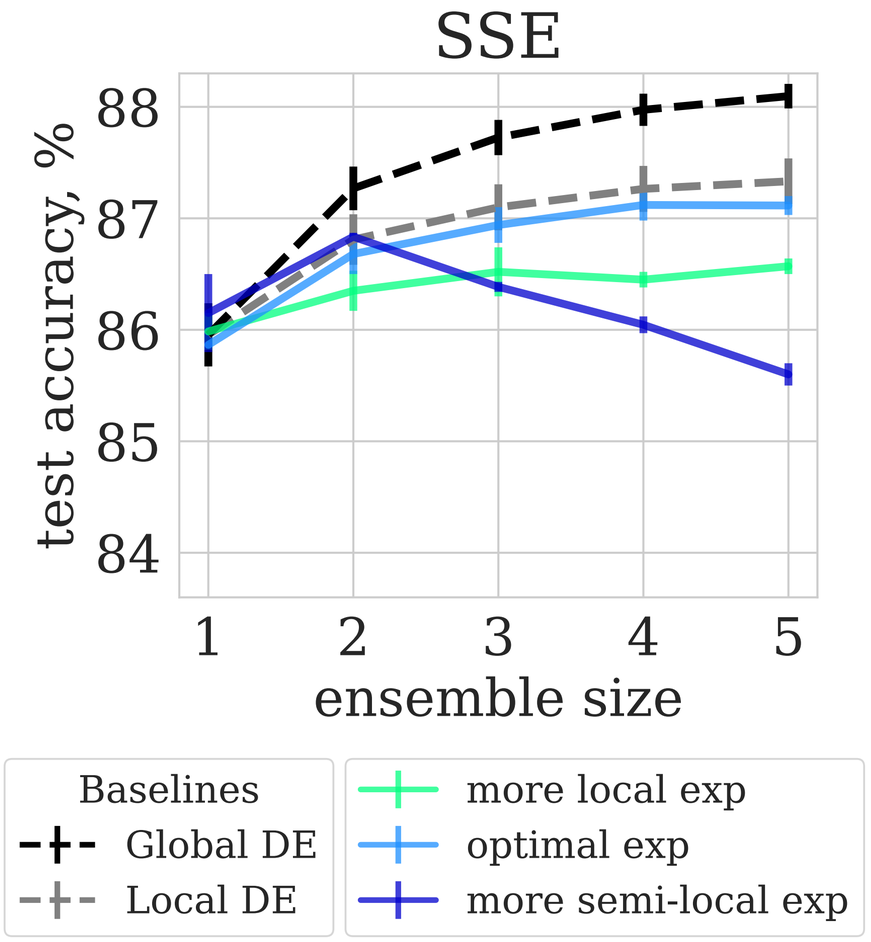
Local and semi-local behavior of SSE



- Low hyperparameters → same basin → **local** behavior
- High hyperparameters → neighboring basins → **semi-local** behavior

Is it better to use SSE in a local or semi-local regime?

SSE results



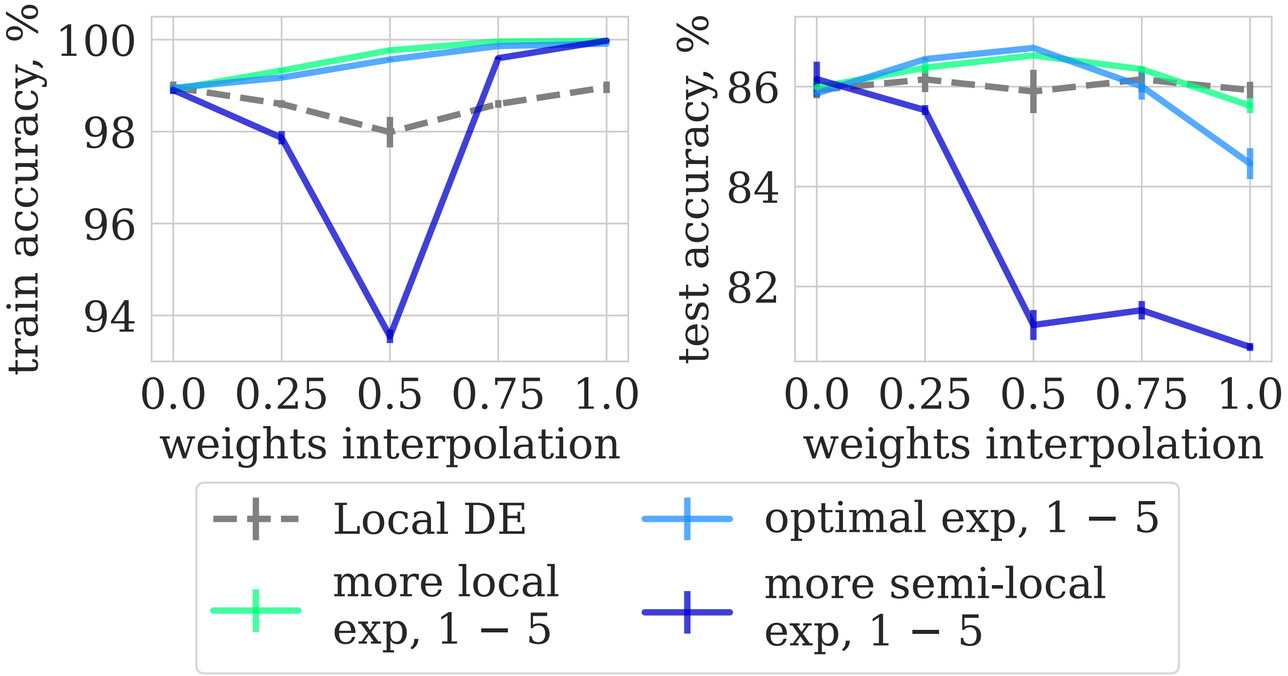
ResNet-50, CIFAR-100,
BYOL self-supervised pre-training.

3 main SSE results:

- **More local SSE** — models are very close, slowly growing ensemble quality
- **Optimal SSE** — more diverse models, quality comparable to Local DE
- **More semi-local SSE** — low quality of ensembles of larger sizes

Sadrtdinov et al., 2023

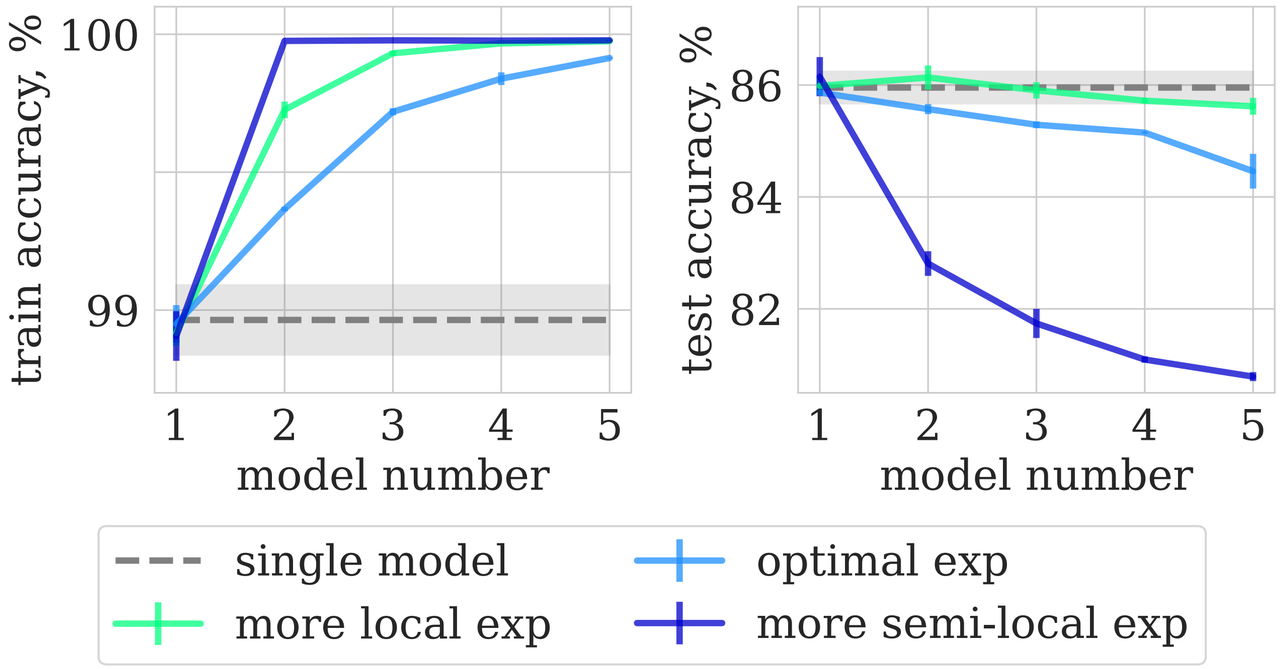
SSE analysis



ResNet-50, CIFAR-100,
BYOL self-supervised pre-training.

- **More local SSE** & **Optimal SSE** — local behavior (no accuracy drop in the middle, same basin)
- **More semi-local SSE** — semi-local behaviour (accuracy drop in the middle, different basins)

SSE analysis



ResNet-50, CIFAR-100,
BYOL self-supervised pre-training.

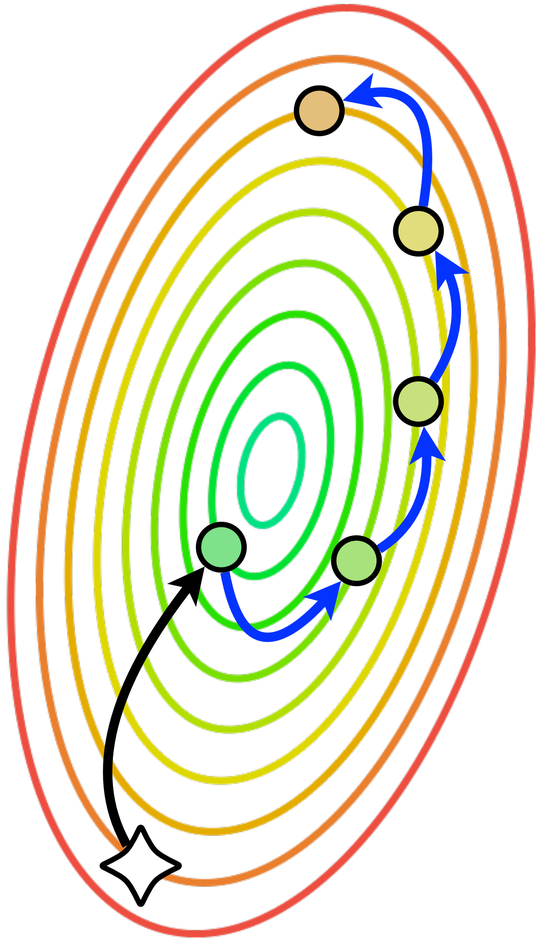
After each cycle:

- Train accuracy ↑
- Test accuracy ↓

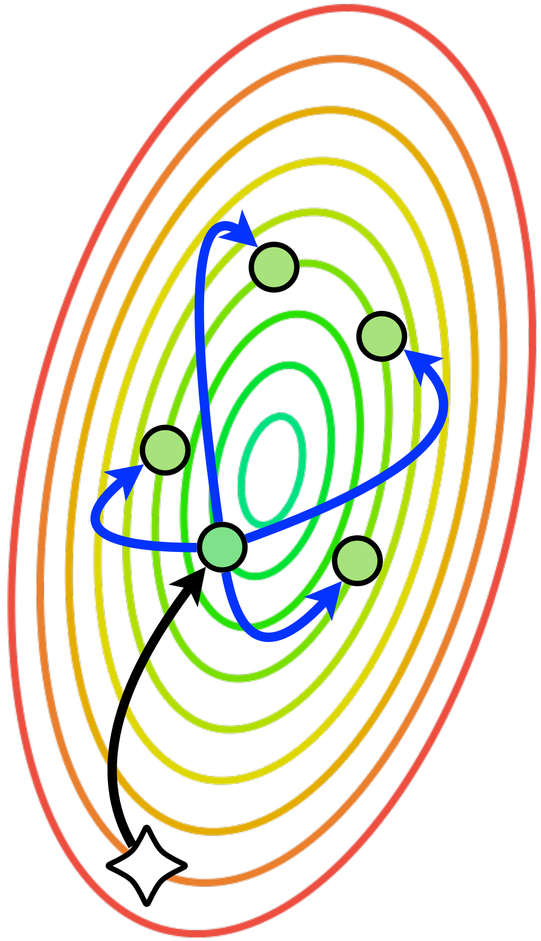
SSE:

- overfits
- goes too far from pre-trained checkpoint
- loses advantages of transfer learning

SSE and StarSSE



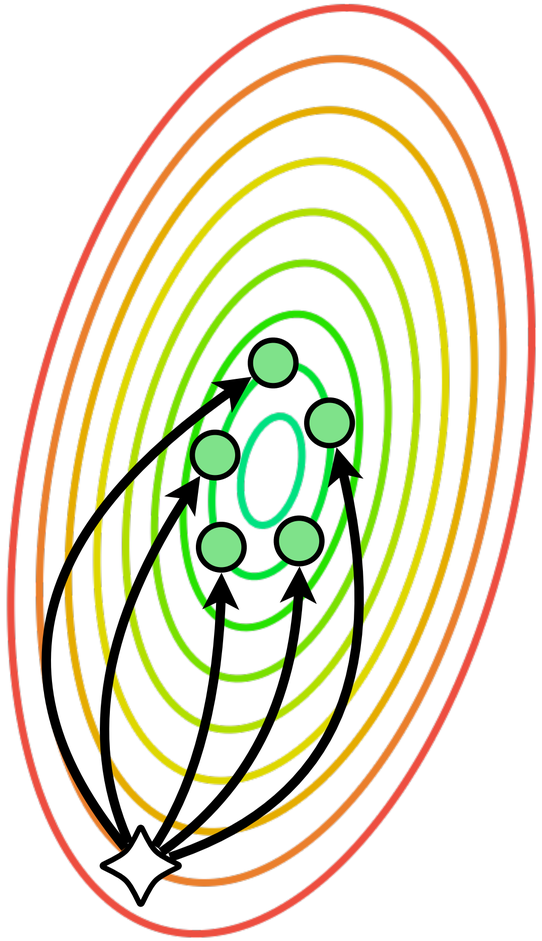
SSE



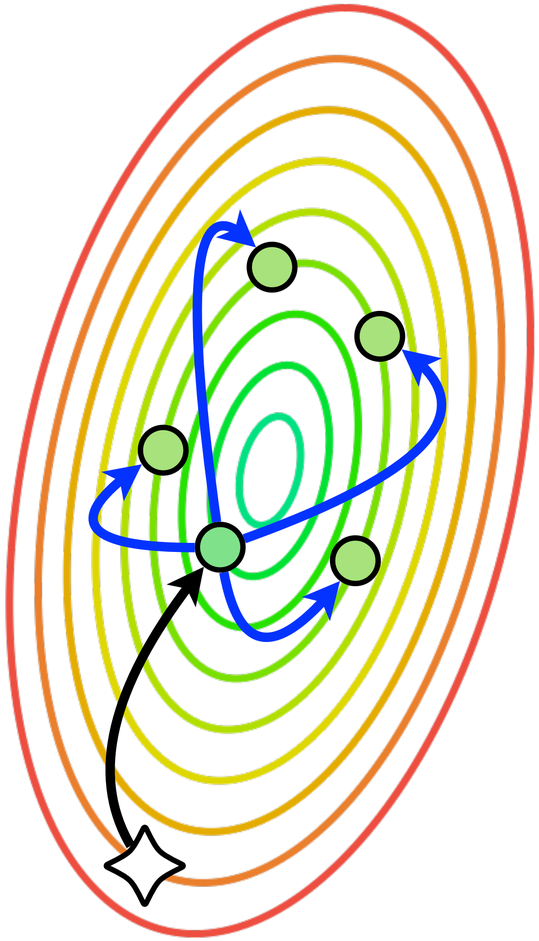
StarSSE

- **Problem:** sequential training → degradation of models quality
- **Solution:** train models in parallel!
- First network trained similarly to SSE
- Rest of models trained in parallel starting from the first network

StarSSE and Local DE



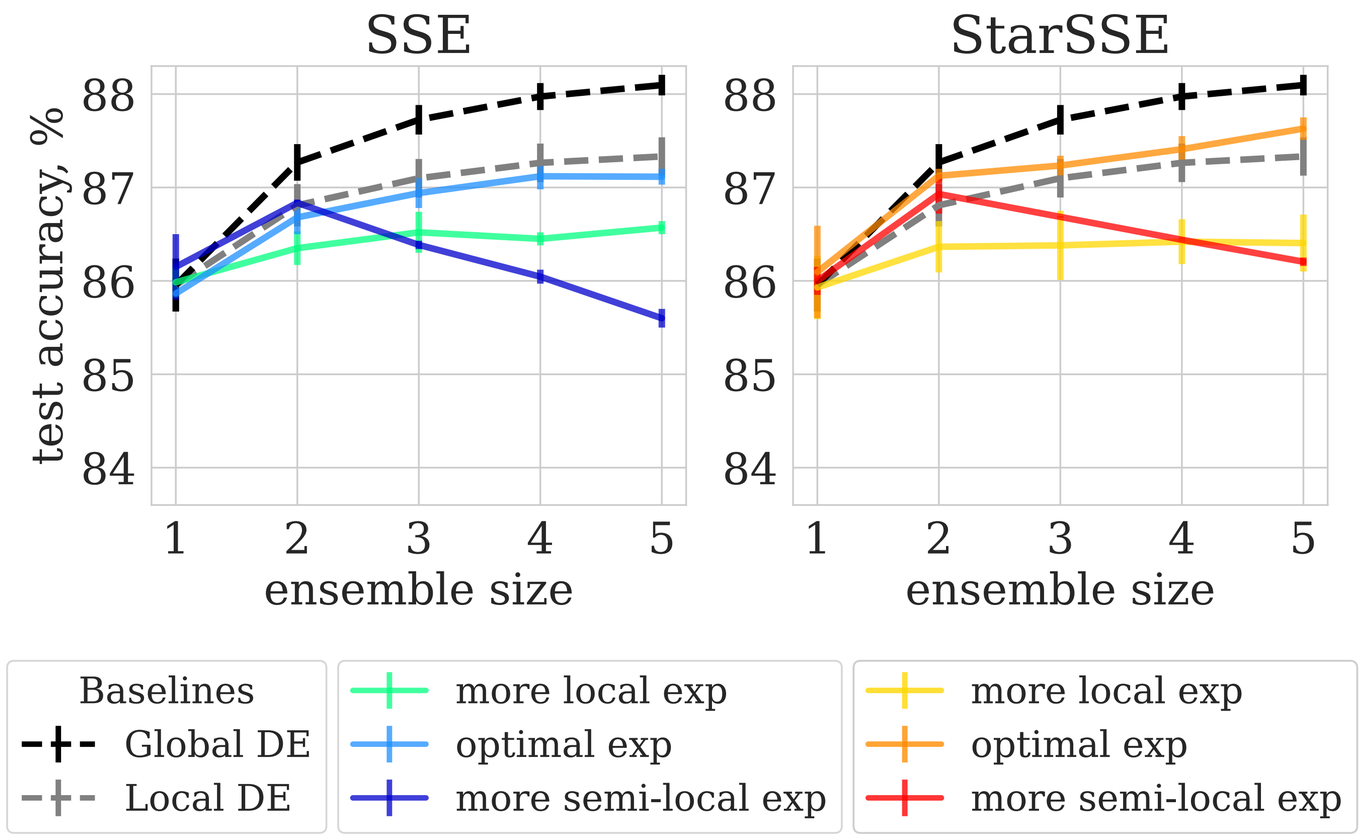
Local DE



StarSSE

- **Local DE:** parallel training from pre-trained checkpoint
- **StarSSE:** parallel training from fine-tuned model
- StarSSE separates **moving to low-loss region** and **pre-train basin exploration**!

StarSSE results

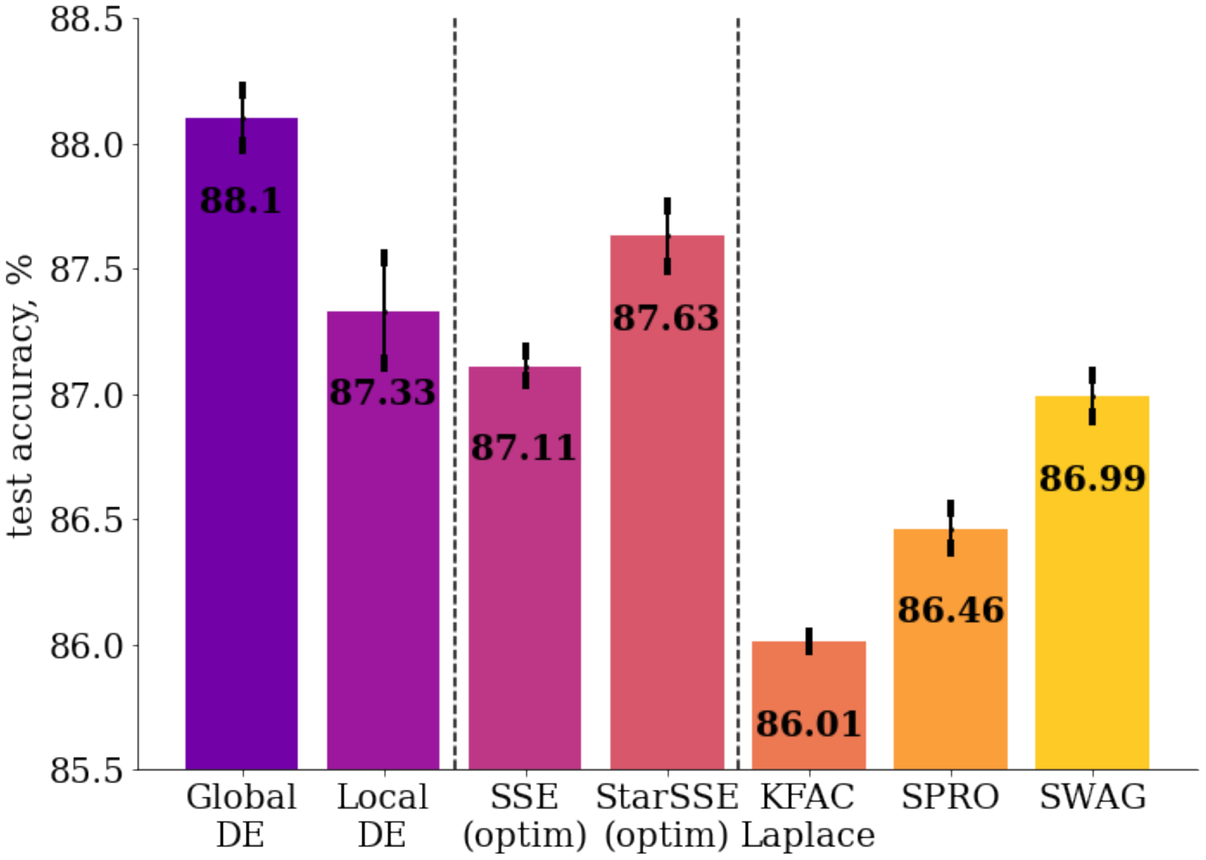


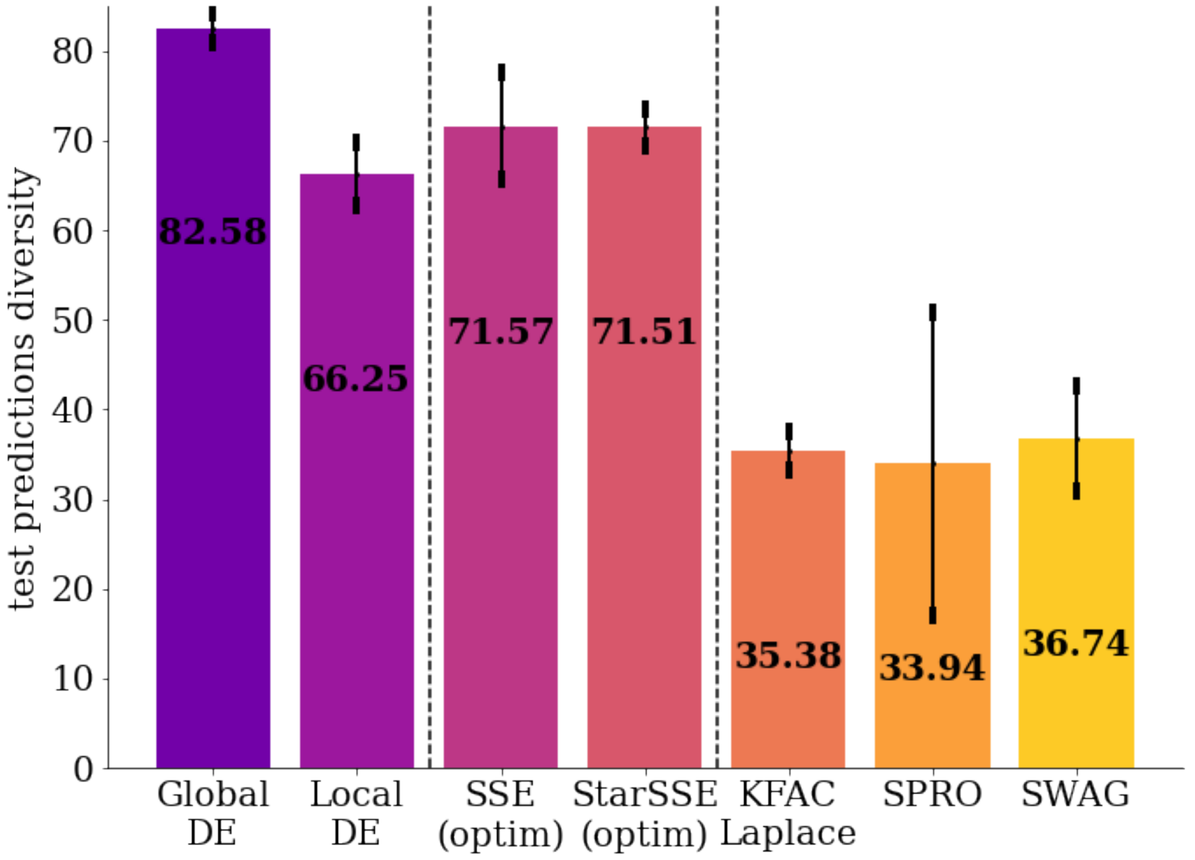
- **Optimal StarSSE** outperforms both **optimal SSE** and **Local DE**
- **Semi-local StarSSE** quality degrades less than **semi-local SSE**

ResNet-50, CIFAR-100, BYOL self-supervised pre-training.

Results

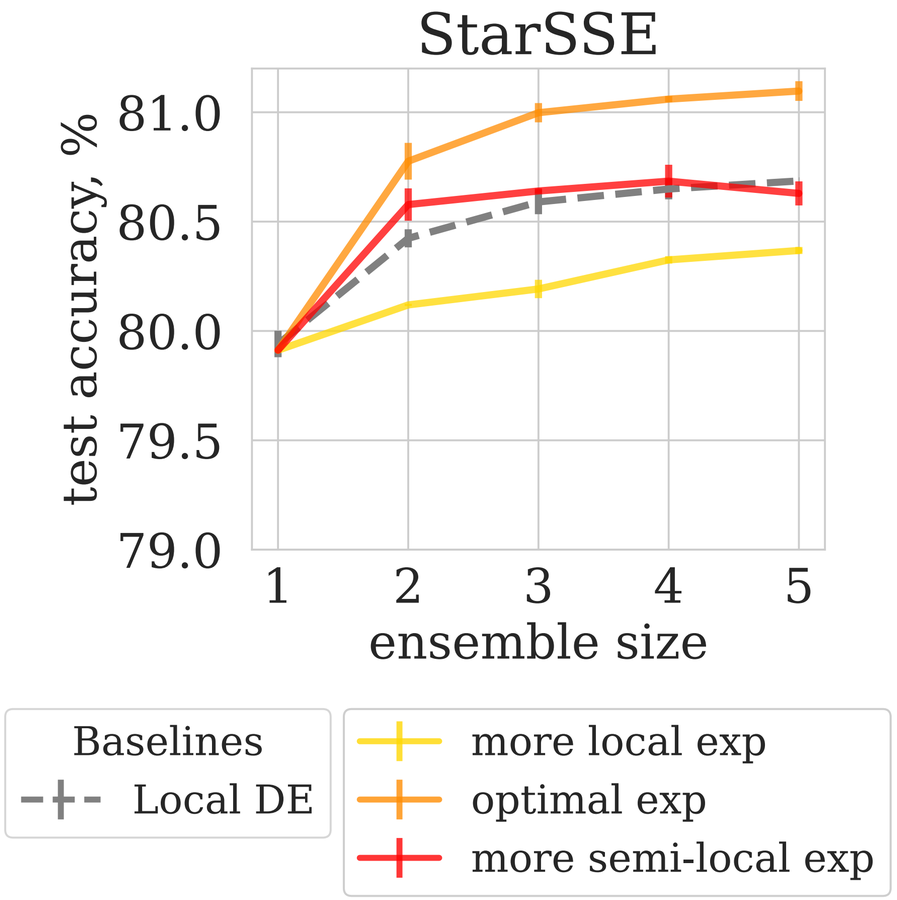
$$diversity = 100 \cdot \mathbb{E}_{m_1 \neq m_2} \frac{\mathbb{E}_{images}[pred_1 \neq pred_2]}{\max(err_1, err_2)}$$

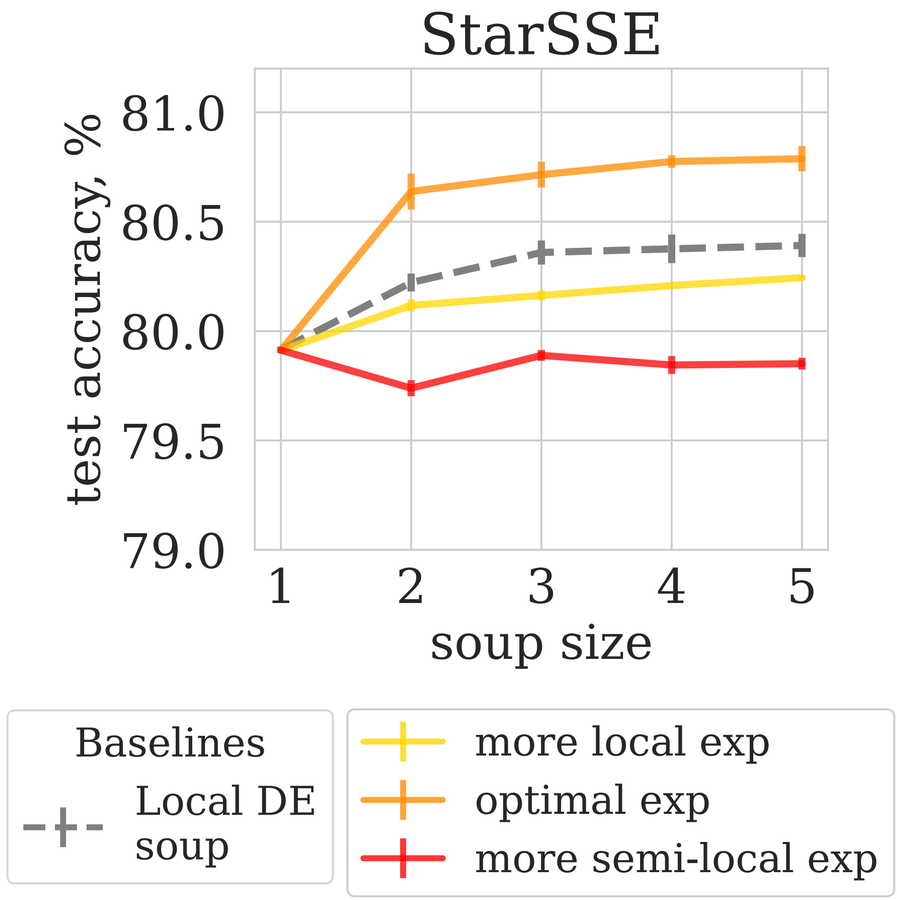




ResNet-50, CIFAR-100, BYOL self-supervised pre-training.

StarSSE results



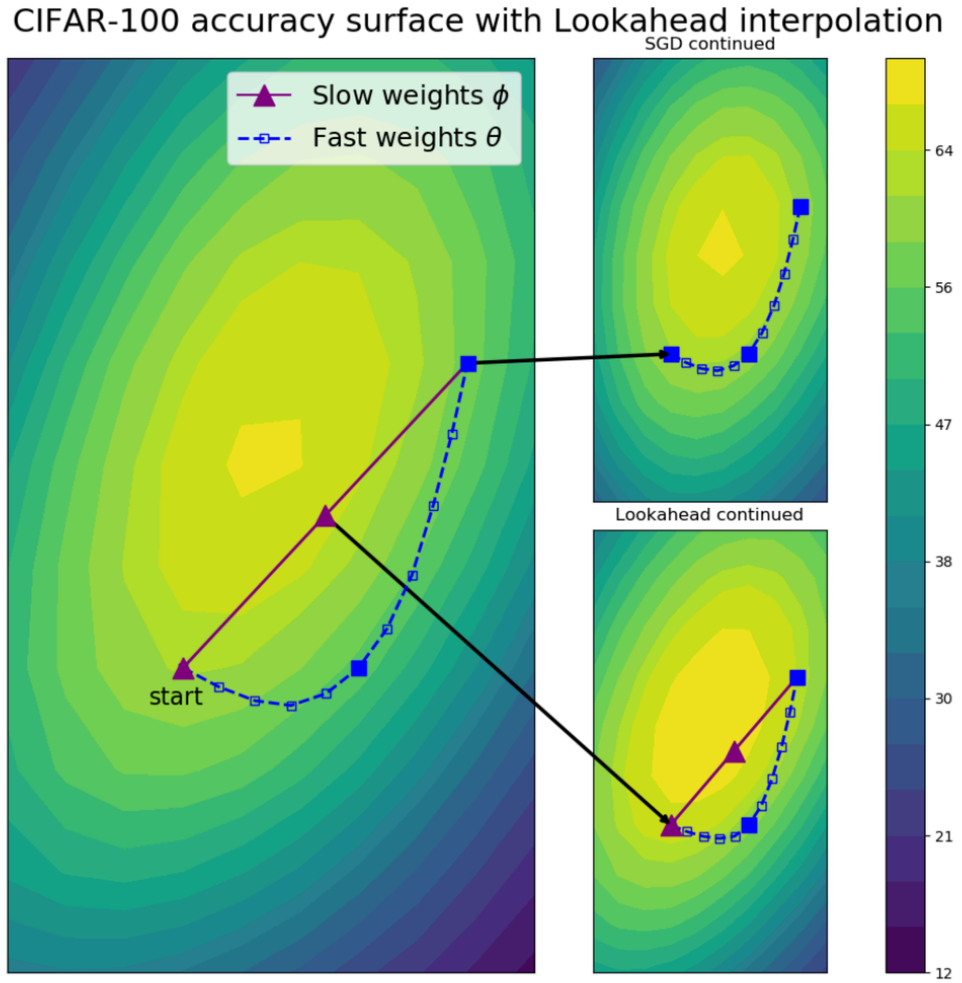


ViT-B/32, ImageNet fine-tuning, CLIP pre-training.

Plan

- Mode connectivity and Linear Mode Connectivity (LMC)
- Weight averaging techniques
  - Stochastic Weight Averaging (SWA)
  - Model soups
- Ensembling methods
  - Deep Ensembles
  - Cyclical methods: SSE, FGE, cSGLD
  - Non-cyclical methods: KFAC-Laplace, SWAG, SPRO
- Ensembles in transfer learning
- **Weight-averaging based optimizers**

Lookahead optimizer



Algorithm 1 Lookahead Optimizer:

Require: Initial parameters ϕ_0, objective function L
Require: Synchronization period k, slow weights step size α, optimizer A
for $t = 1, 2, \ldots$ **do**
  Synchronize parameters $\theta_{t,0} \leftarrow \phi_{t-1}$
  for $i = 1, 2, \ldots, k$ **do**
    sample minibatch of data $d \sim \mathcal{D}$
    $\theta_{t,i} \leftarrow \theta_{t,i-1} + A(L, \theta_{t,i-1}, d)$
  end for
  Perform outer update $\phi_t \leftarrow \phi_{t-1} + \alpha(\theta_{t,k} - \phi_{t-1})$
end for
return parameters ϕ

Zhang et al., 2019

Lookaround optimizer

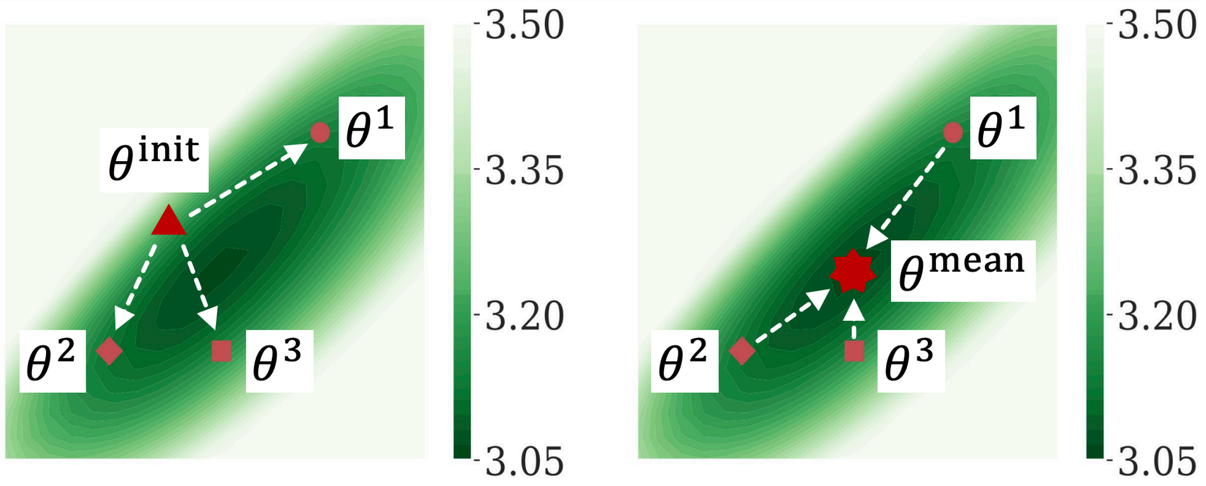


Figure 1: Test set loss landscape. (Left) Around step for diversity. (Right) Average step for locality.

Algorithm 1 Lookaround Optimizer.

Require: Initial parameters ϕ_0, objective function $\mathcal{L}$, data augmentation list AUG of size d, synchronization period k, optimizer A, dataset $\mathcal{D}$, numbers of training epochs E.

for *epoch* $= 1$ to E **do**
  Synchronize parameters
  for $j = 1, 2, \ldots, d$ **do**
    $\theta_{t,j,0} \leftarrow \phi_{t-1}$
  end for
  # Around Step: Independent model training.
  for $i = 1, 2, \ldots, k$ **do**
    sample minibatch of data $B \sim \mathcal{D}$
    for $j = 1, 2, \ldots, d$ **do**
      $\theta_{t,j,i} \leftarrow \theta_{t,j,i-1} + A(\mathcal{L}, \theta_{t,j,i-1}, \text{AUG}_j(B))$
    end for
  end for
  # Average Step: Weight averaging.
  Compute average weight $\theta_{t,*,k} \leftarrow \frac{1}{d} \sum_{j=1}^{d} \theta_{t,j,k}$
  Perform update $\phi_t \leftarrow \theta_{t,*,k}$
end for
return parameters ϕ

Zhang et al., 2023

Lookahead & Lookaround

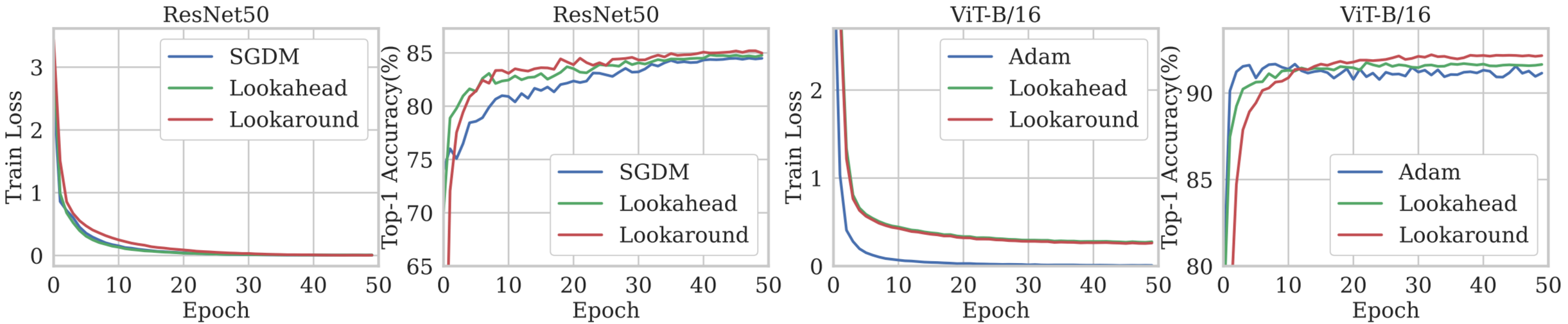
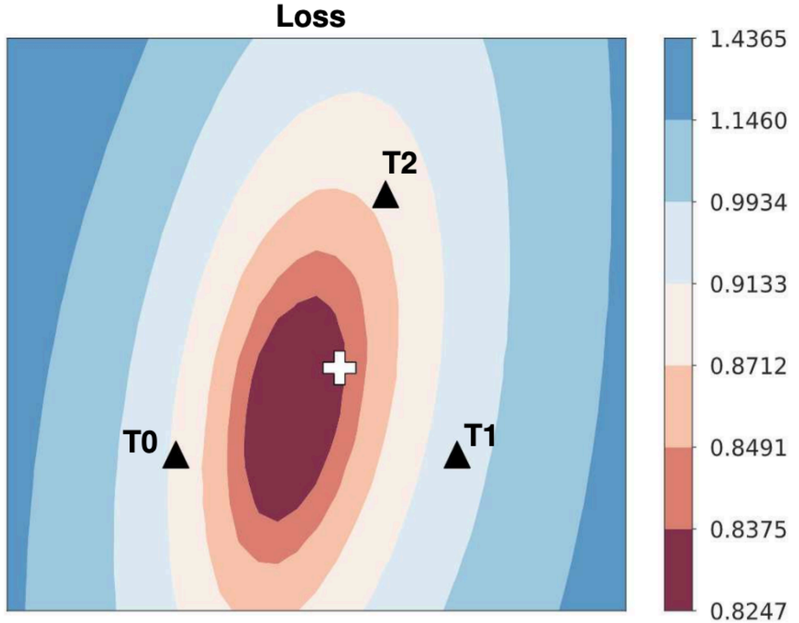
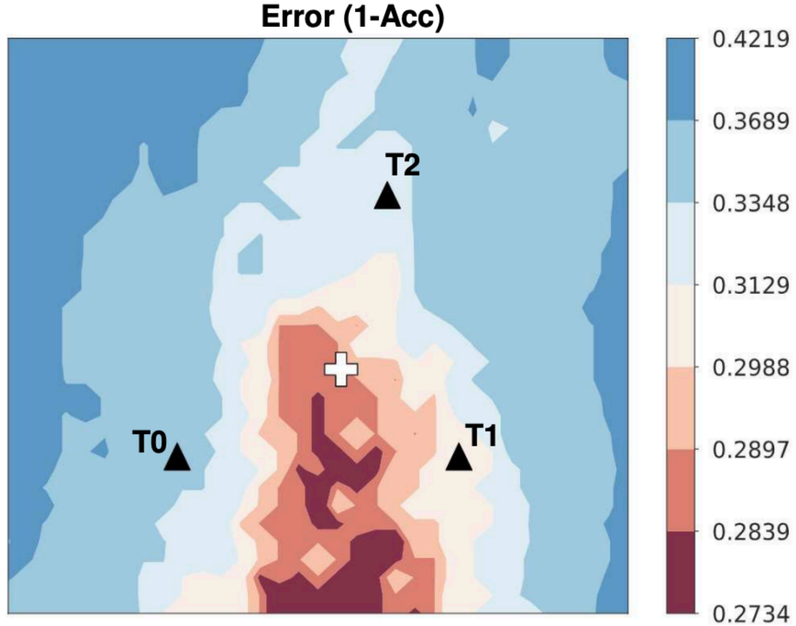


Figure 4: Training loss and Top-1 accuracy curves of ResNet50 (left) and ViT-B/16 (right) on CIFAR100 under different optimization methods.

WATT: Weight Average Test-Time Adaptation

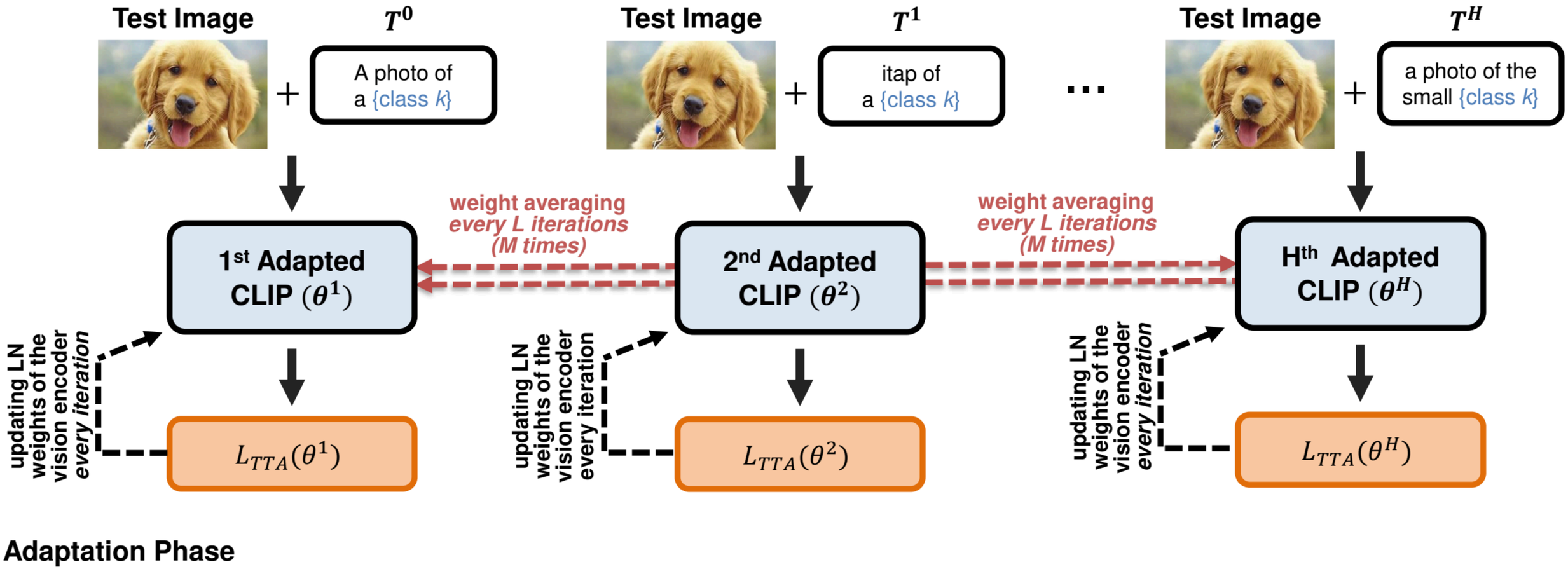




	Template
T^0:	"a photo of a {class k}"
T^1:	"itap of a {class k}"
T^2:	"a bad photo of the {class k}"
T^3:	"a origami {class k}"
T^4:	"a photo of the large {class k}"
T^5:	"a {class k} in a video game"
T^6:	"art of the {class k}"
T^7:	"a photo of the small {class k}"

Osowiechi et al., 2024

WATT: Weight Average Test-Time Adaptation



WATT: Weight Average Test-Time Adaptation

$$p_{ik} = \frac{\exp\left(\cos(\mathbf{z}_i^v, \mathbf{z}_k^t)/\tau\right)}{\sum_j \exp\left(\cos(\mathbf{z}_i^v, \mathbf{z}_j^t)/\tau\right)}, \quad \cos(\mathbf{z}, \mathbf{z}') = \frac{\mathbf{z}^\top \mathbf{z}'}{\|\mathbf{z}\|_2 \cdot \|\mathbf{z}'\|_2}$$

Vision embeddings similarity

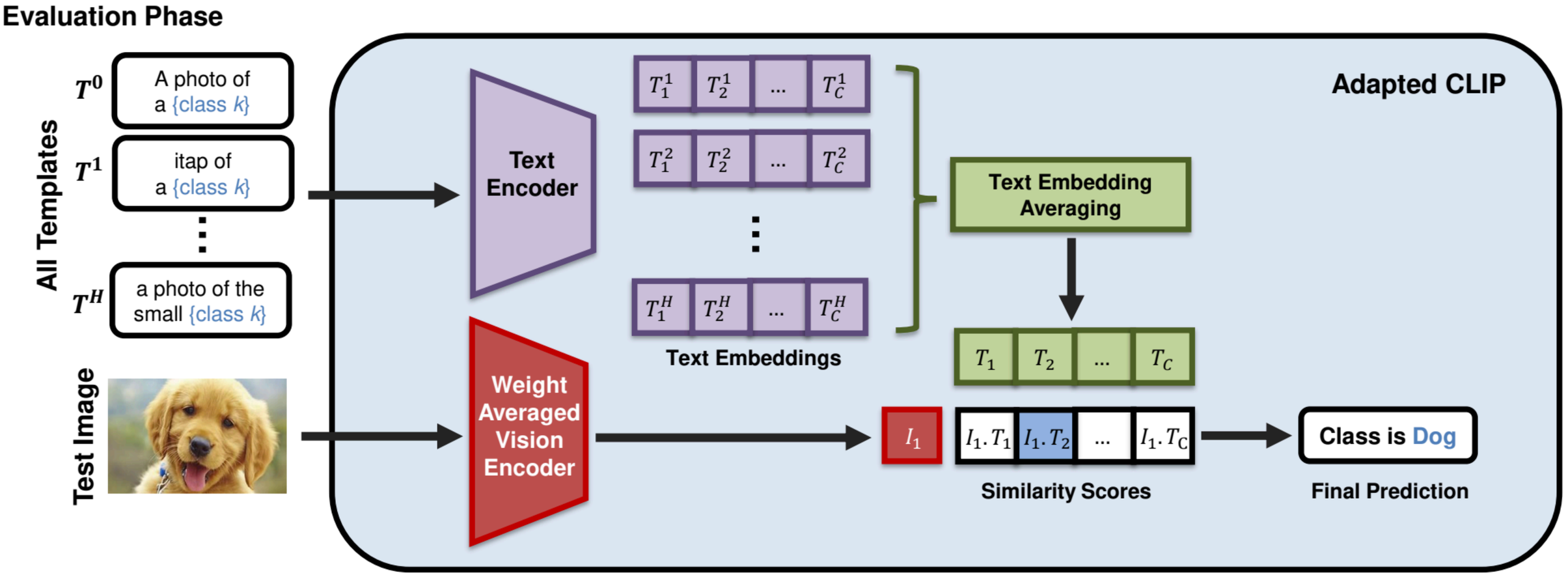
Text embeddings similarity

$$\mathbf{S}^v = \mathbf{Z}^v(\mathbf{Z}^v)^\top \qquad \mathbf{S}^t = \mathbf{Z}^t(\mathbf{Z}^t)^\top$$

$$\mathbf{Q} = \mathrm{softmax}\big((\mathbf{S}^v + \mathbf{S}^t)/2\tau\big)$$

$$\mathcal{L}_{\mathrm{TTA}}(\theta) = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{B} q_{ij} \log p_{ij}$$

WATT: Weight Average Test-Time Adaptation



WATT: Weight Average Test-Time Adaptation

Dataset	Text avg.	Output avg.	Weight avg. (ours)		
			(after 10 iter)×1	(after 1 iter)×10	(after 2 iter)×5
CIFAR-10	90.58 ±0.03	90.90 ±0.03	91.08 ±0.06	**91.39 ±0.14**	91.05 ±0.06
CIFAR-10.1	85.78 ±0.25	86.77 ±0.08	86.85 ±0.18	**88.02 ±0.18**	86.98 ±0.31
CIFAR-10-C	71.41	72.60	72.66	73.66	**73.82**
CIFAR-100	69.46 ±0.13	70.32 ±0.1	70.3 ±0.11	**70.85 ±0.08**	70.74 ±0.20
CIFAR-100-C	41.37	42.68	42.24	45.32	**45.57**

Table 5: Accuracy (%) obtained with different averaging strategies.

WATT: Weight Average Test-Time Adaptation

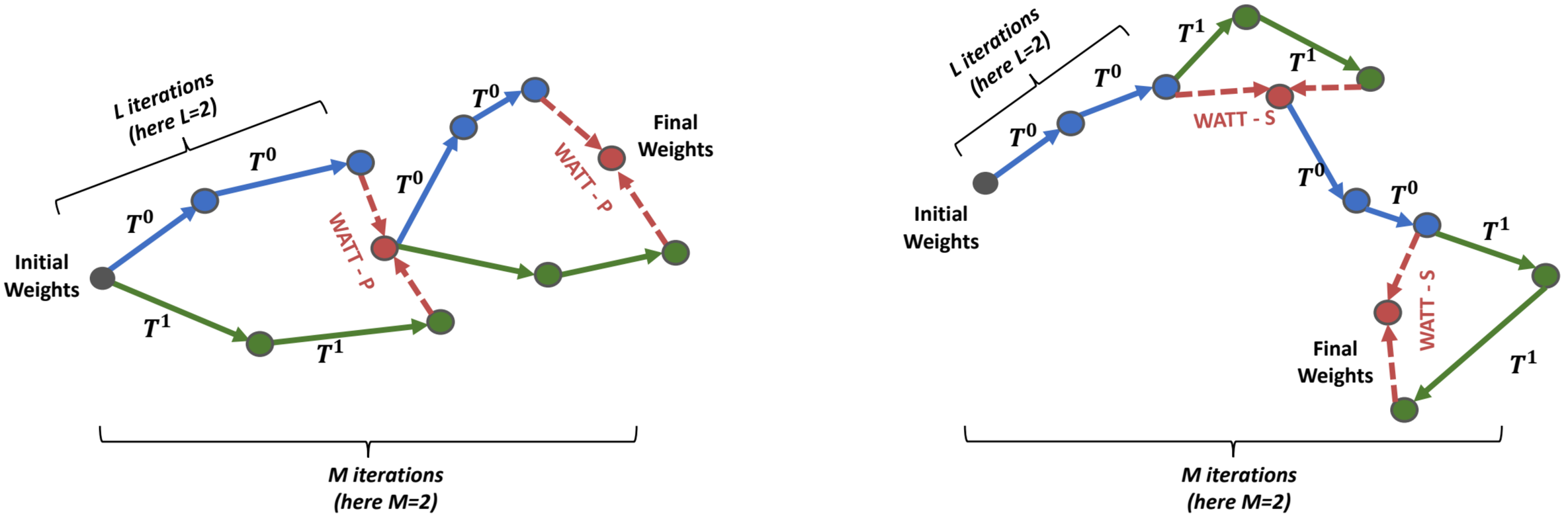


Figure 3: Visual comparison of the Parallel **(left)** and Sequential **(right)** approaches for multi-template weight averaging during adaptation.

WATT: Weight Average Test-Time Adaptation

Dataset		CLIP	TENT	TPT (BS=32)	CLIPArTT	WATT-P	WATT-S
CIFAR-10		88.74	**91.69** ±0.10	88.06 ±0.06	90.04 ±0.13	91.41 ±0.17	91.05 ±0.06
CIFAR-10.1		83.25	87.60 ±0.45	81.80 ±0.27	86.35 ±0.27	**87.78** ±0.05	86.98 ±0.31
CIFAR-10-C		59.22	67.56	56.80	71.17	72.83	**73.82**
CIFAR-100		61.68	69.74 ±0.16	63.78 ±0.28	69.79 ±0.04	70.38 ±0.14	**70.74** ±0.20
CIFAR-100-C	Gaussian Noise	14.80	14.38 ±0.14	14.03 ±0.10	25.32 ±0.14	31.28 ±0.03	**32.07** ±0.23
	Shot noise	16.03	17.34 ±0.27	15.25 ±0.17	27.90 ±0.05	33.44 ±0.11	**34.36** ±0.11
	Impulse Noise	13.85	10.03 ±0.13	13.01 ±0.13	25.62 ±0.09	29.40 ±0.11	**30.33** ±0.03
	Defocus blur	36.74	49.05 ±0.07	37.60 ±0.17	49.88 ±0.23	52.32 ±0.28	**52.99** ±0.16
	Glass blur	14.19	3.71 ±0.07	16.41 ±0.02	27.89 ±0.03	31.20 ±0.12	**32.15** ±0.30
	Motion blur	36.14	46.62 ±0.27	37.52 ±0.23	47.93 ±0.14	49.72 ±0.15	**50.53** ±0.12
	Zoom blur	40.24	51.84 ±0.15	42.99 ±0.11	52.70 ±0.06	54.72 ±0.04	**55.30** ±0.22
	Snow	38.95	46.71 ±0.21	42.35 ±0.13	49.72 ±0.01	51.79 ±0.04	**52.77** ±0.15
	Frost	40.56	44.90 ±0.27	43.31 ±0.14	49.63 ±0.12	53.04 ±0.08	**53.79** ±0.31
	Fog	38.00	47.31 ±0.04	38.81 ±0.17	48.77 ±0.04	50.78 ±0.24	**51.49** ±0.21
	Brightness	48.18	60.58 ±0.18	50.23 ±0.11	61.27 ±0.08	62.65 ±0.25	**63.57** ±0.21
	Contrast	29.53	45.90 ±0.11	28.09 ±0.09	48.55 ±0.24	51.34 ±0.10	**52.76** ±0.27
	Elastic transform	26.33	33.09 ±0.08	28.12 ±0.15	37.45 ±0.08	39.97 ±0.06	**40.90** ±0.43
	Pixelate	21.98	26.47 ±0.09	20.43 ±0.14	33.88 ±0.14	39.59 ±0.09	**40.97** ±0.16
	JPEG compression	25.91	29.89 ±0.07	28.82 ±0.09	36.07 ±0.32	38.99 ±0.16	**39.59** ±0.08
	Mean	29.43	35.19	30.46	41.51	44.68	**45.57**