Designing Neural Network Architecture for Deep Reinforcement Learning

Belgium-Netherlands Reinforcement Learning (BeNeRL) Seminar Series

Hojoon Lee

DAVIAN Data and Visual Analytics Lab

KAIST AI Graduate School of AI

Outline

1. Scaling Laws of Deep Neural Network

2. SimBa: Simplicity Bias for Scaling Up Parameters in Deep RL

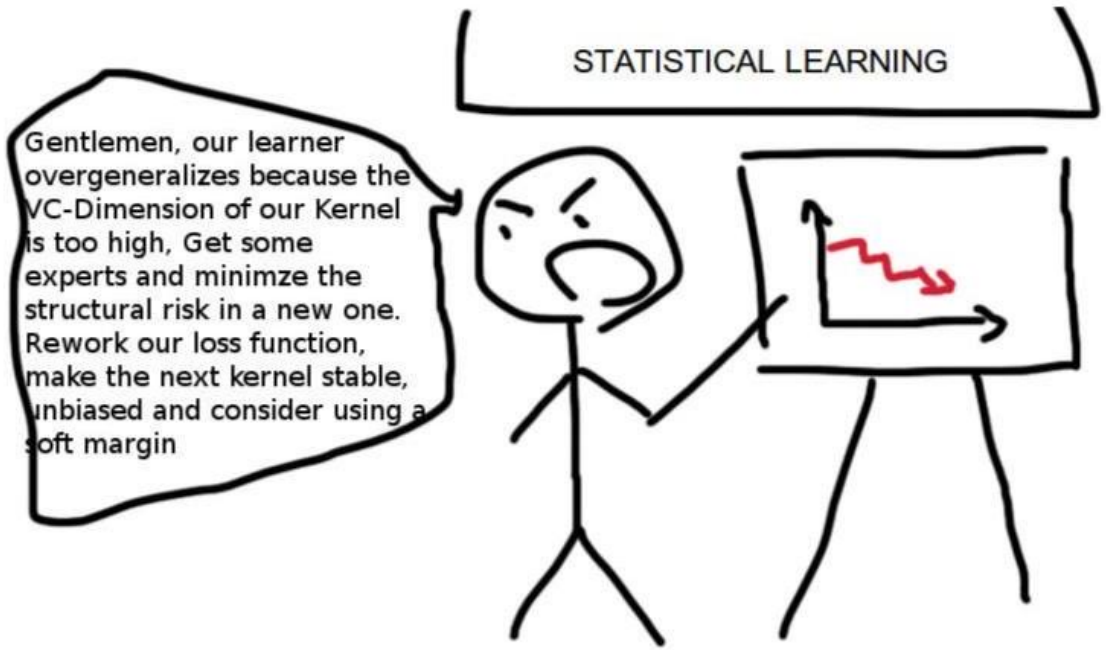
   Hojoon Lee*, Dongyoon Hwang*, Donghu Kim, Hyunseung Kim, Jet Tai, Kaushik Subramanian, Peter Wurman, Jaegul Choo, Peter Stone, Takuma Seno, arXiv 2024

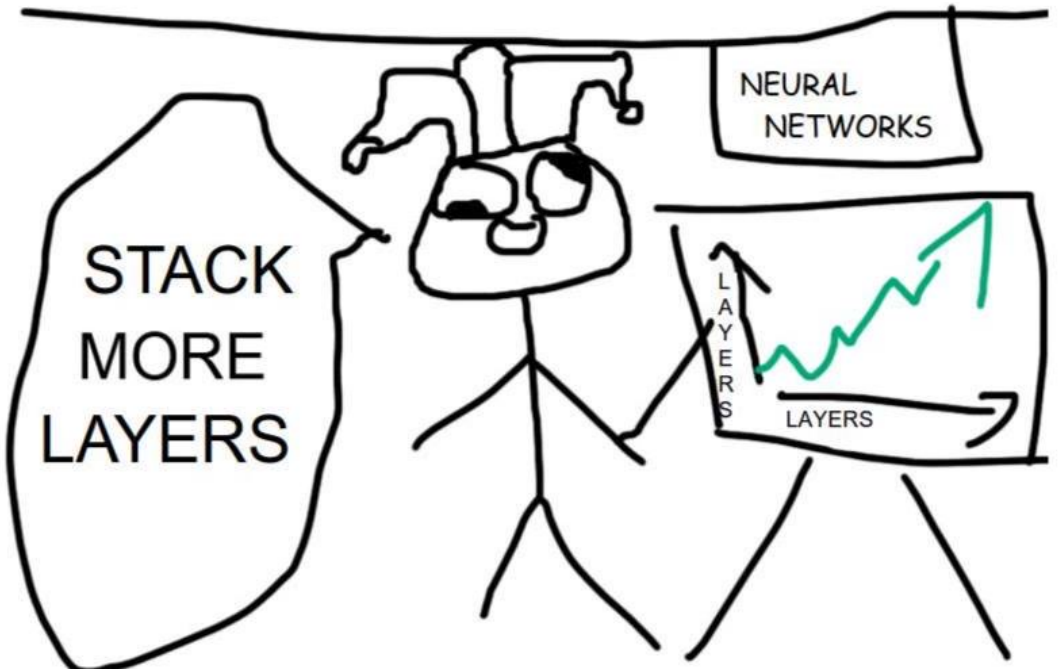
3. Slow and Steady Wins the Race: Maintaining Plasticity with Hare and Tortoise Networks

   Hojoon Lee, Hyeonseo Cho, Hyunseung Kim, Donghu Kim, Dugki Min, Jaegul Choo, Clare Lyle, ICML 2024

4. Future Research Direction

Who Would Win?

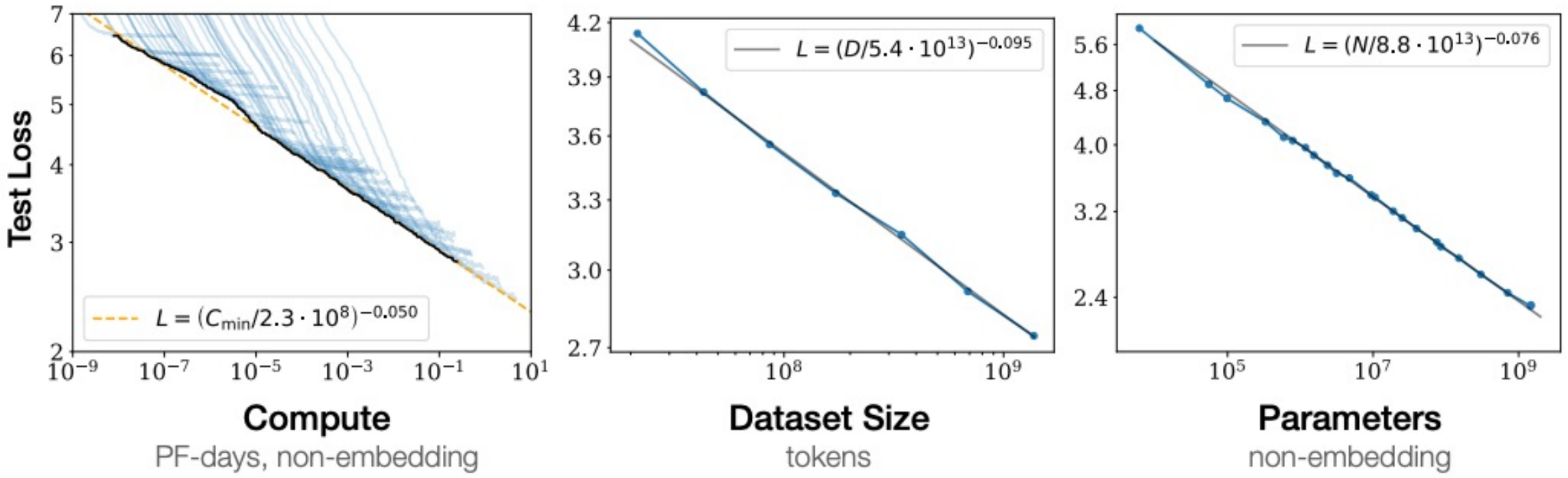




https://www.reddit.com/r/ProgrammerHumor/comments/8c1i45/stack_more_layers/

Scaling Laws of Deep Neural Network

Empirically, network performance improves as we scale up the compute, dataset, and model size.



[1] Scaling Laws for Neural Language Models, Kaplan et al, arXiv 2020.

Double Descent Phenomenon

[1] Deep Double Descent: Where Bigger Models and More Data Hurt, Nakkiran et al, arXiv 2019.

Why Neural Networks do not Overfit?

A bunch of optimization & architectural components

- Optimization [1,2,3]
  - Gradient Noise in Stochastic Gradient Descent.
  - Weight Decay.
  - Cross Entropy Loss.
- Architecture [4]
  - ReLU-like activations.
  - Small weights / activations.
  - Batch / Layer / Instance … Normalizations.
  - Residual connections.

[1] On the Generalization Benefit of Noise in Stochastic Gradient Descent, Smith et al, ICML 2020.
[2] SGD and Weight Decay Secretly Minimize the Rank of Your Neural Network, Galanti et al, arXiv 2024.
[3] Cross-Entropy Loss and Low Rank Features Have Responsibility for Adversarial Examples, Nar et al, arXv 2019.
[4] Neural Redshift: Random Networks are not Random Functions, Teney et al, CVPR 2024.

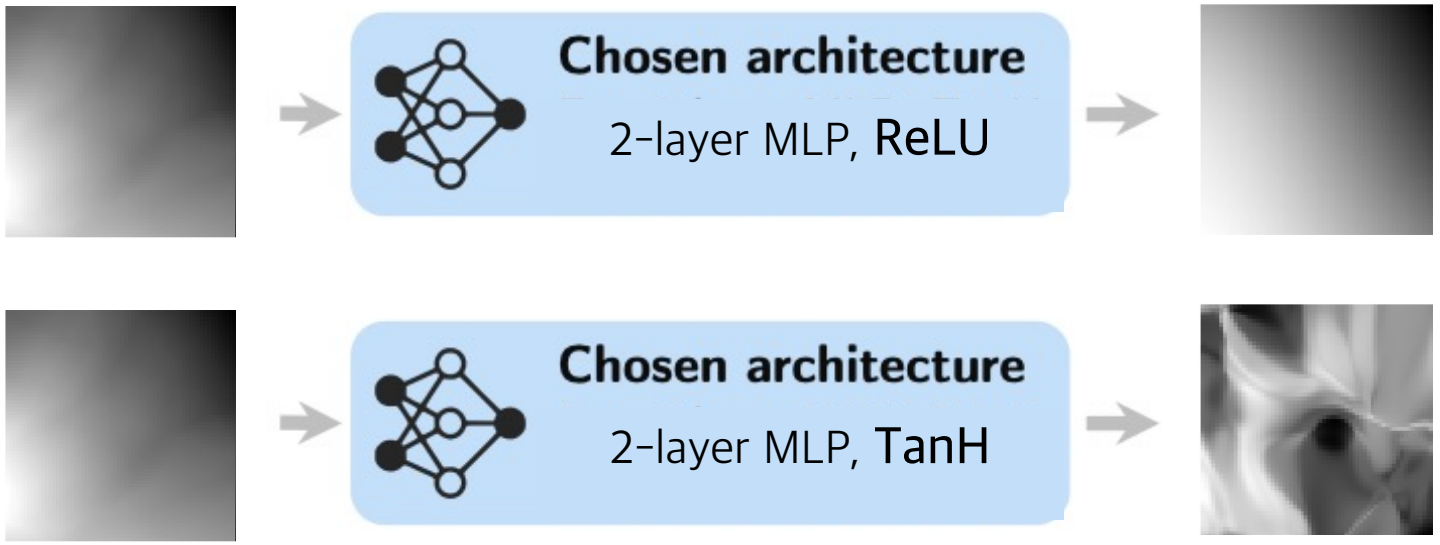
Why Neural Networks do not Overfit?

Formally speaking, Simplicity Bias [1]

Modern networks contain architectural components that lead them to favor simpler functions [2]

e.g., ReLU activation

e.g., low frequency

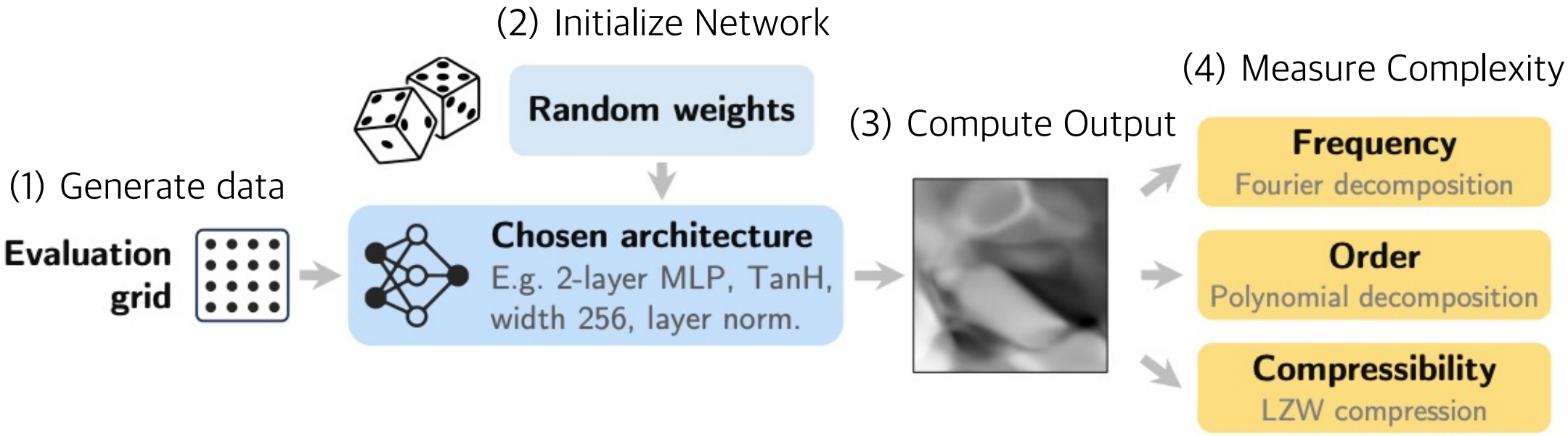


[1] Neural Networks are a-priori biased towards low entropy functions., Mingard et al, arXiv 2019.

[2] Neural Redshift: Random Networks are not Random Functions, Teney et al, CVPR 2024.

Why Neural Networks do not Overfit?

How to measure the Simplicity Bias? [1]



E.g., Architectures with their output represented by lower polynomials have a higher simplicity bias.

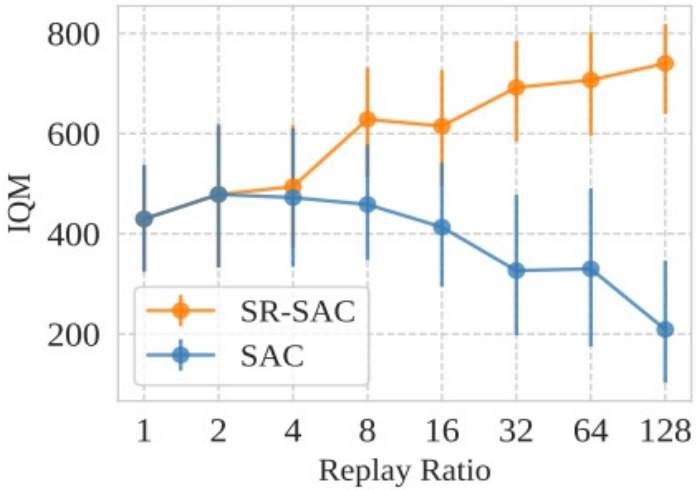
[1] Neural Redshift: Random Networks are not Random Functions, Teney et al, CVPR 2024.

Does Scaling Laws Hold in Deep RL?

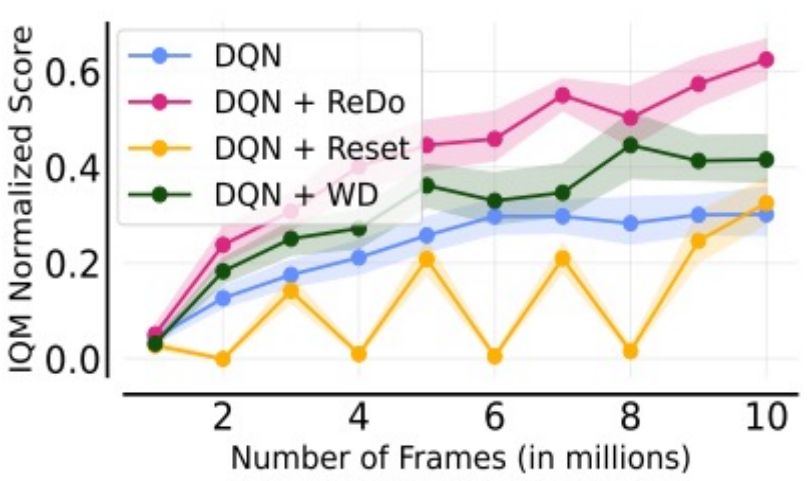
Yes and No

With periodic network reinitialization, the scaling laws hold.

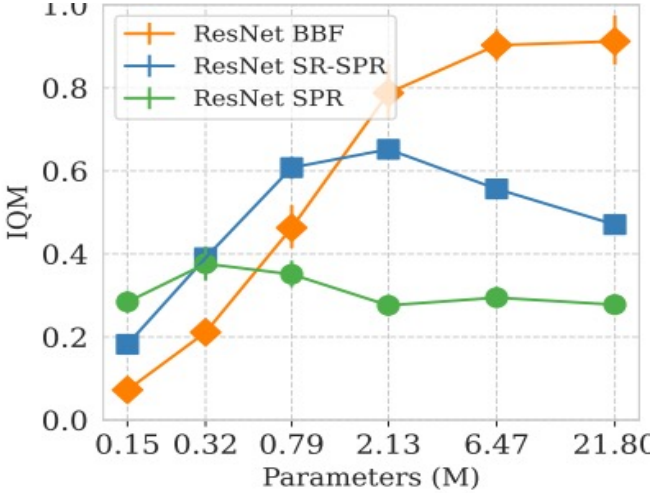
Without periodic network reinitialization, the scaling laws does not hold.



Compute [1]



Dataset [2]



Parameters [3]

[1] Sample-Efficient Reinforcement Learning by Breaking the Replay Ratio Barrier, D'Oro et al, ICLR 2023.

[2] Bigger, Better, Faster: Human-level Atari with human-level efficiency, Schwarzer et al, ICML 2023.

[3] The Dormant Neuron Phenomenon in Deep Reinforcement Learning, Sokar et al, ICML 2023.

Why Scaling Laws Do not Hold in Deep RL?

1. Simplicity Bias

In supervised learning, simplicity bias is a key factor driving scaling laws.

Maybe the RL community is not fully harnessing the simplicity bias.

2. Non-stationary Optimization

Unlike SL, RL is trained from non-stationary data distributions.

The network can overfit to earlier data and fail to optimize to new data.

Network reinitialization transforms this setup into a stationary optimization.

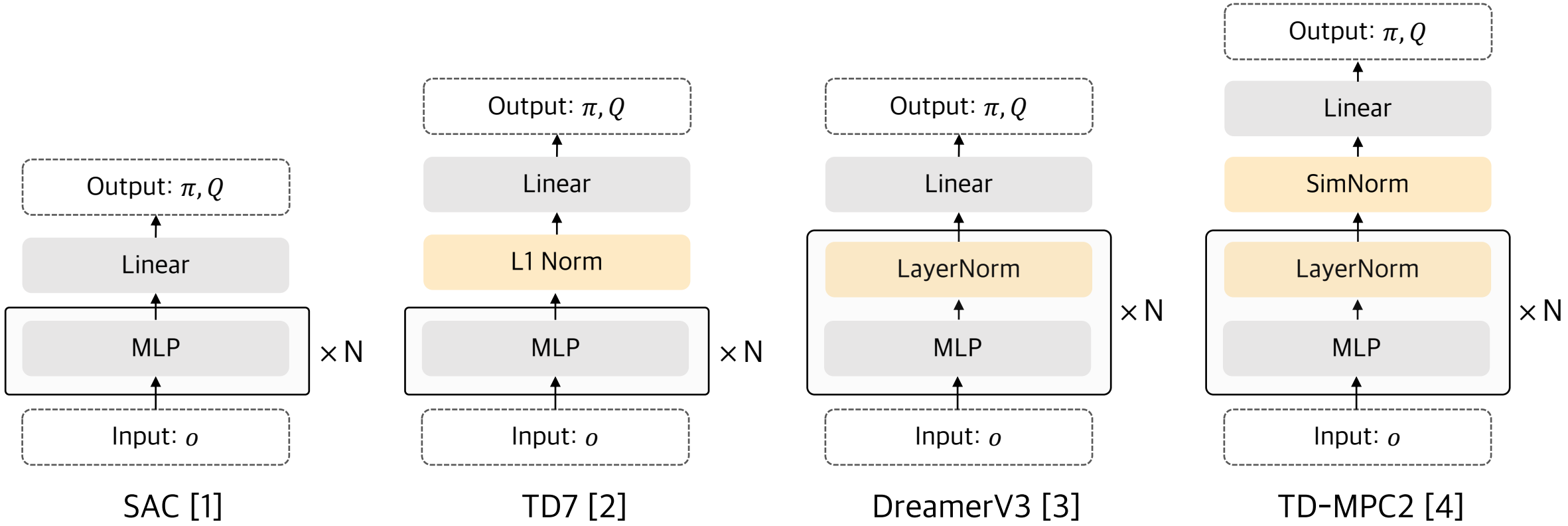
SimBa: Simplicity Bias for Scaling Up Parameters in Deep Reinforcement Learning

Hojoon Lee*, Dongyoon Hwang*, Donghu Kim, Hyunseung Kim, Jet Tai,
Kaushik Subramanian, Peter Wurman, Jaegul Choo, Peter Stone, Takuma Seno

KAIST AI Graduate School of AI
Sony AI
TEXAS The University of Texas at Austin

A Standard Network Architecture in Deep RL

*World model components have been omitted for an ease of understanding



[1] Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, Haarnoja et al, ICML 2018.

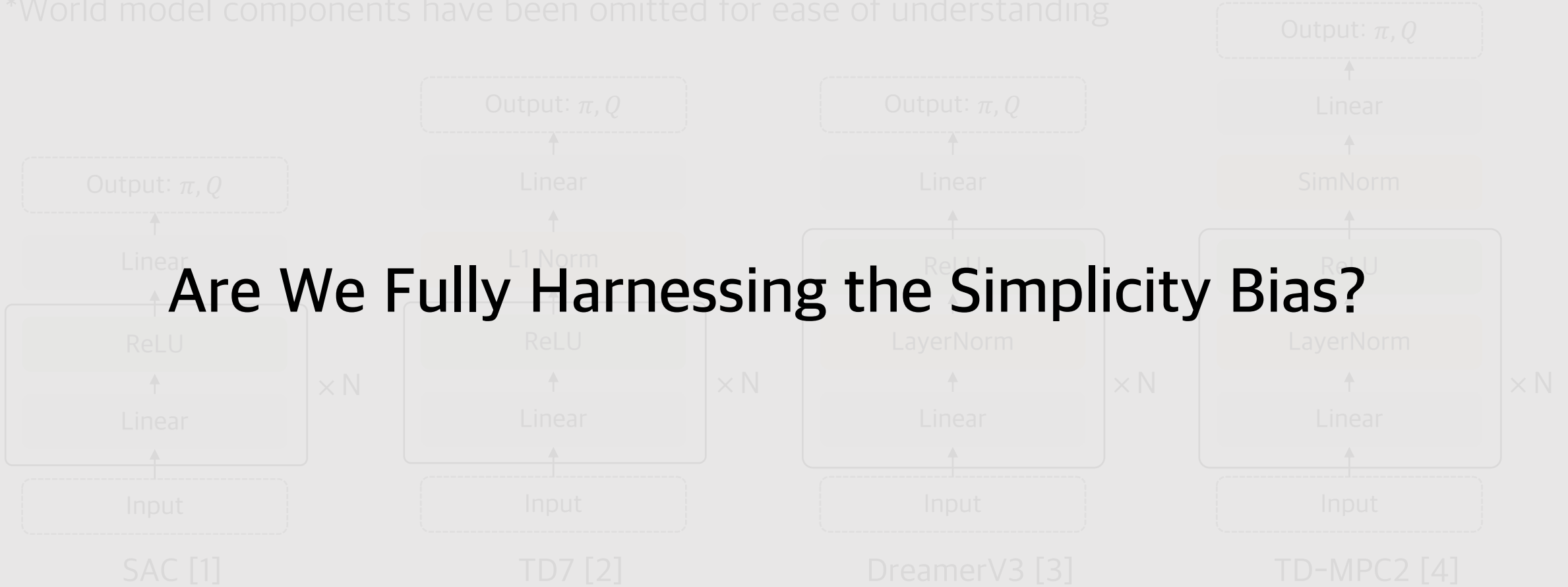
[2] For SALE: State-Action Representation Learning for Deep Reinforcement Learning, Fujimoto et al, NeurIPS 2023.

[3] DreamerV3: Mastering Diverse Domains through World Models, Hafner et al, arXiv 2023.

[4] TD-MPC2: Scalable, Robust World Models for Continuous Control, Hansen et al, ICLR 2024.

A Standard Network Architecture in Deep RL: MLP

*World model components have been omitted for ease of understanding



Are We Fully Harnessing the Simplicity Bias?

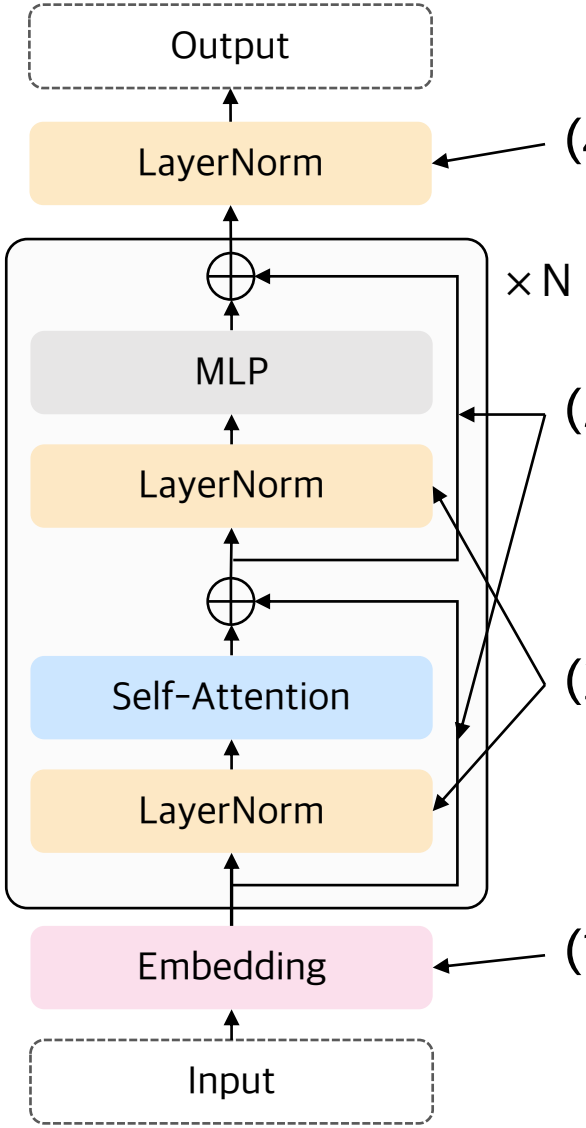
[1] Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, Haarnoja et al, ICML 2018.

[2] For SALE: State-Action Representation Learning for Deep Reinforcement Learning, Fujimoto et al, NeurIPS 2023.

[3] DreamerV3: Mastering Diverse Domains through World Models, Hafner et al, arXiv 2023.

[4] TD-MPC2: Scalable, Robust World Models for Continuous Control, Hansen et al, ICLR 2024.

Demystifying Simplicity Bias in Transformer



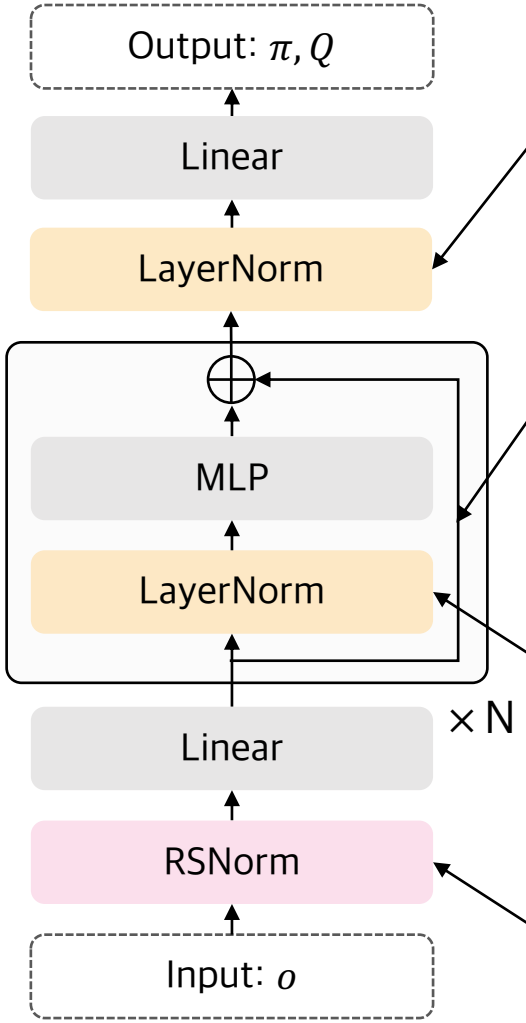
(4) **Post LayerNorm**: Normalizes aggregated features before the classifier.

(3) **Residual Connection**: Provide a direct linear pathway from input to output.

(2) **Pre LayerNorm**: Normalizes feature before applying non-linear transform.

(1) **Embedding**: Embeds inputs with small, scaled weights.

SimBa: Injecting Simplicity Bias into Deep RL Architecture



(4) **Post LayerNorm**: Normalizes aggregated features before the classifier.

(3) **Residual Connection**: Provide a direct linear pathway from input to output.

(2) **Pre LayerNorm**: Normalizes feature before applying non-linear transform.

(1) **RSNorm**: Normalizes each input dimension to a normal distribution.

SimBa: Injecting Simplicity Bias into Deep RL Architecture

Running Statistics Normalization (RSNorm)

Normalizes each input dimension to a normal distribution by running statistics of input dimension.

Input: $o_t \in \mathbb{R}^d$, Running mean: $\mu_t \in \mathbb{R}^d$, Running variance: $\sigma_t \in \mathbb{R}^d$.

$$\delta_t \leftarrow o_t + \mu_{t-1}$$

$$\mu_t \leftarrow \mu_{t-1} + \frac{1}{t}\delta_t$$

$$o_t^2 \leftarrow \frac{t-1}{t}\left(\sigma_{t-1}^2 + \frac{1}{t}\delta_t^2\right)$$

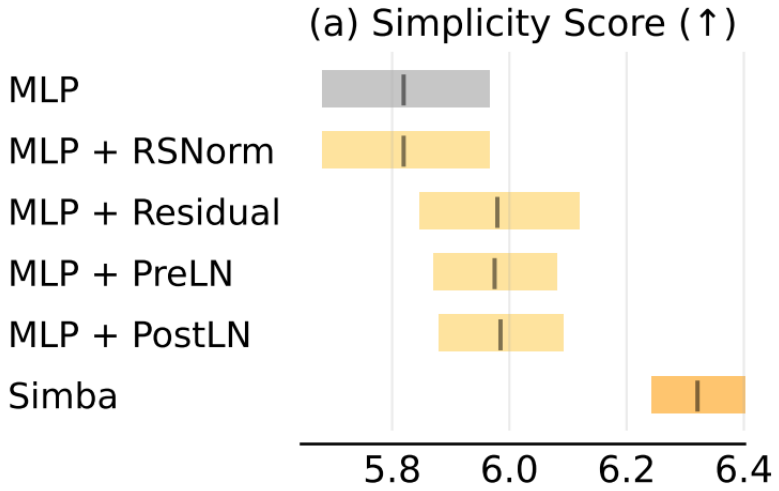
$$\overline{o_t} = \mathrm{RSNorm}(o_t) = \frac{\mathrm{o_t} - \mu_t}{\sqrt{\sigma_t^2 + \epsilon}}$$

SimBa: Injecting Simplicity Bias into Deep RL Architecture

Why RSNorm?

- **LayerNorm**: Struggles with disproportionately large values in certain dimensions.
- **BatchNorm**: Unstable updates due to non-stationary mini-batch statistics.
- **Batch ReNorm [1]**: A good alternative, but requires additional hyperparameter, momentum.

Architecture Analysis



Adding each component increases the Simplicity Bias

*RSNorm is the only exception.

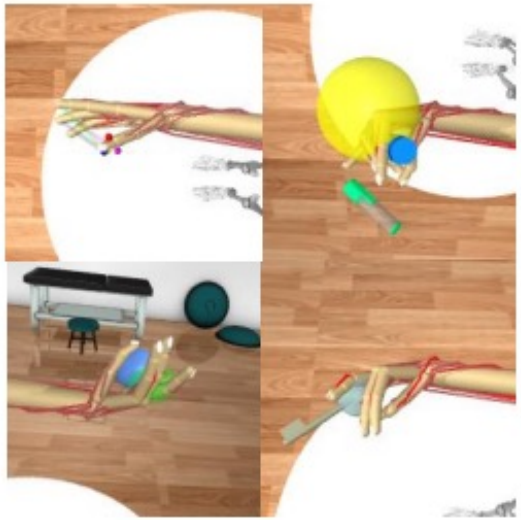
*Likely because we used a uniformly distributed dataset for this analysis.

[1] CrossQ: Batch Normalization in Deep Reinforcement Learning for Greater Sample Efficiency and Simplicity, Bhatt et al, ICLR 2024.

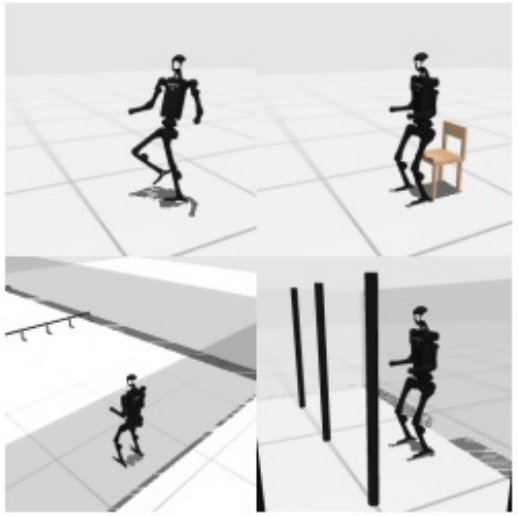
Experiments

Benchmarks

DMC [1]



MyoSuite [2]



Humanoid Bench [3]

Craftax [4]

[1] DeepMind Control Suite, Tassa et al, arXiv 2018.

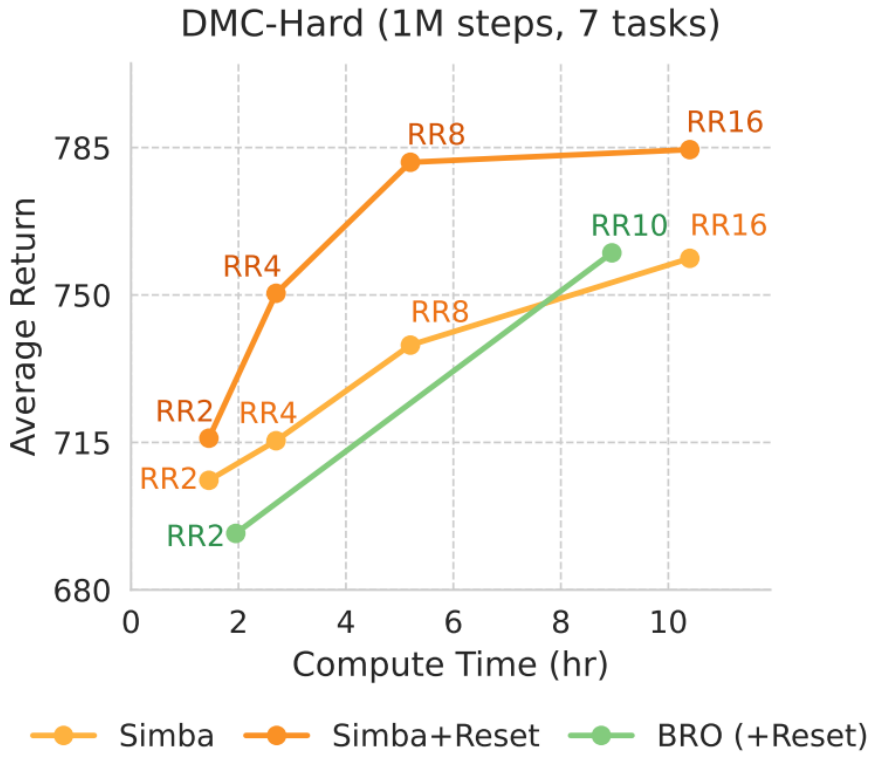
[2] MyoSuite -- A contact-rich simulation suite for musculoskeletal motor control, Caggiano et al, ICML 2022.

[3] HumanoidBench: Simulated Humanoid Benchmark for Whole-Body Locomotion and Manipulation, Sferazza et al, RSS 2024.

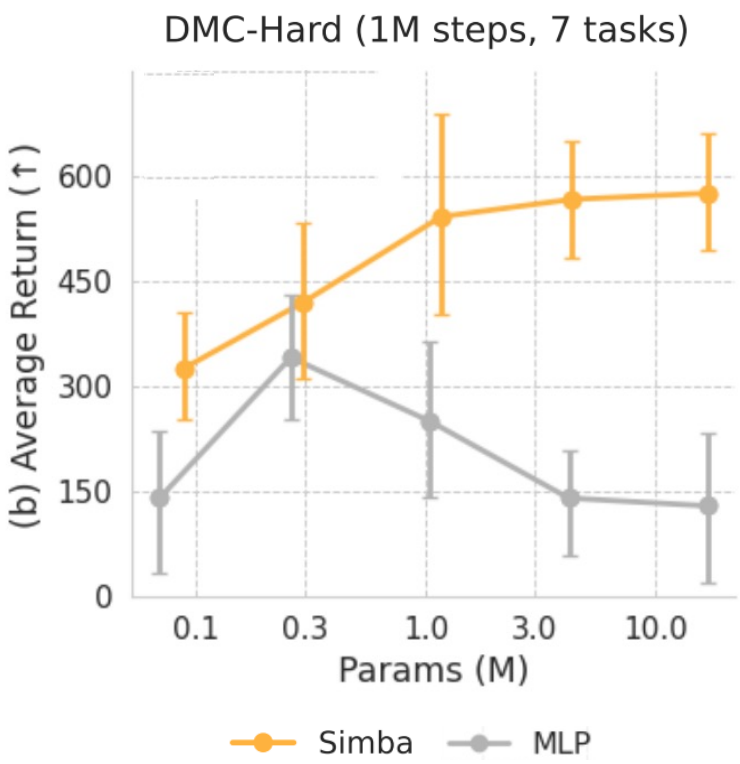
[4] Craftax: A Lightning-Fast Benchmark for Open-Ended Reinforcement Learning, Matthews et al, ICML 2024.

Experiments

When using SimBa, Scaling Laws hold without using a periodic network reinitialization



Compute

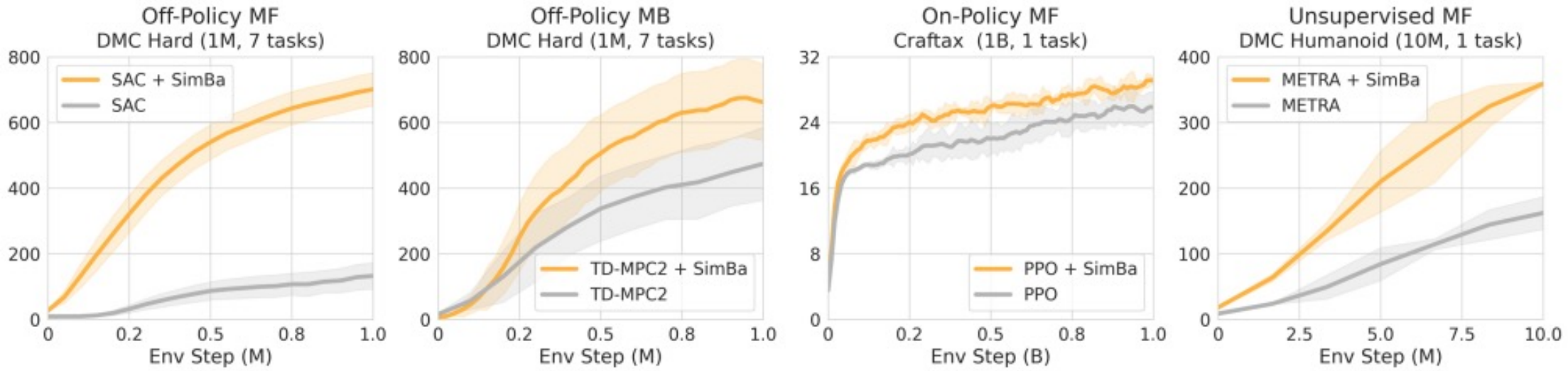


Parameters

[1] Bigger, Regularized, Optimistic: scaling for compute and sample-efficient continuous control, Nauman et al, NeurIPS 2024.

Experiments

Simply replacing the architecture from MLP to SimBa yields huge improvements in various setups



SAC [1] TD-MPC2 [2] PPO [3] METRA [4]

[1] Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, Haarnoja et al, ICML 2018.

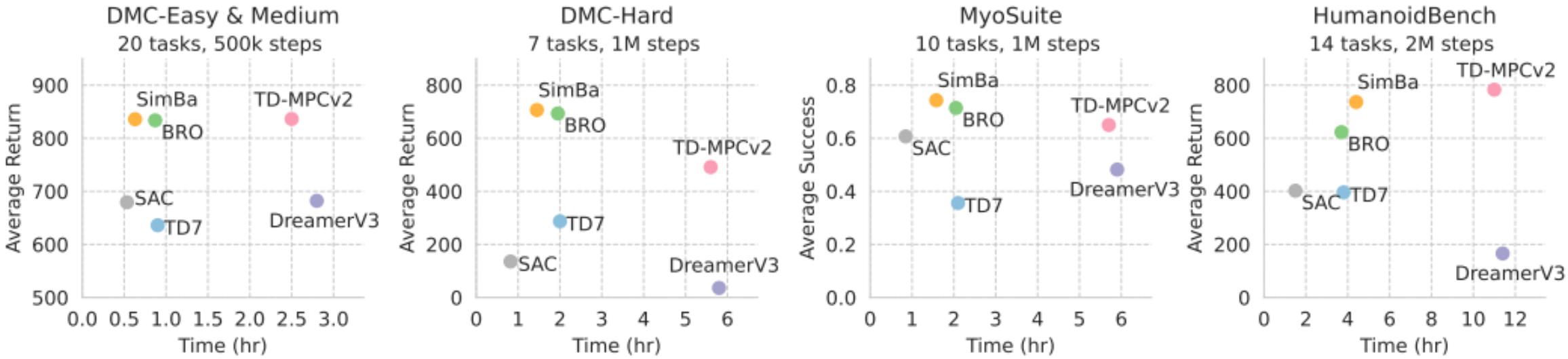
[2] TD-MPC2: Scalable, Robust World Models for Continuous Control, Hansen et al, ICLR 2024.

[3] Proximal Policy Optimization Algorithms for Continuous Control, Schulman et al, arXiv 2017.

[4] METRA: Scalable Unsupervised RL with Metric-Aware Abstraction, Park et al, ICLR 2024.

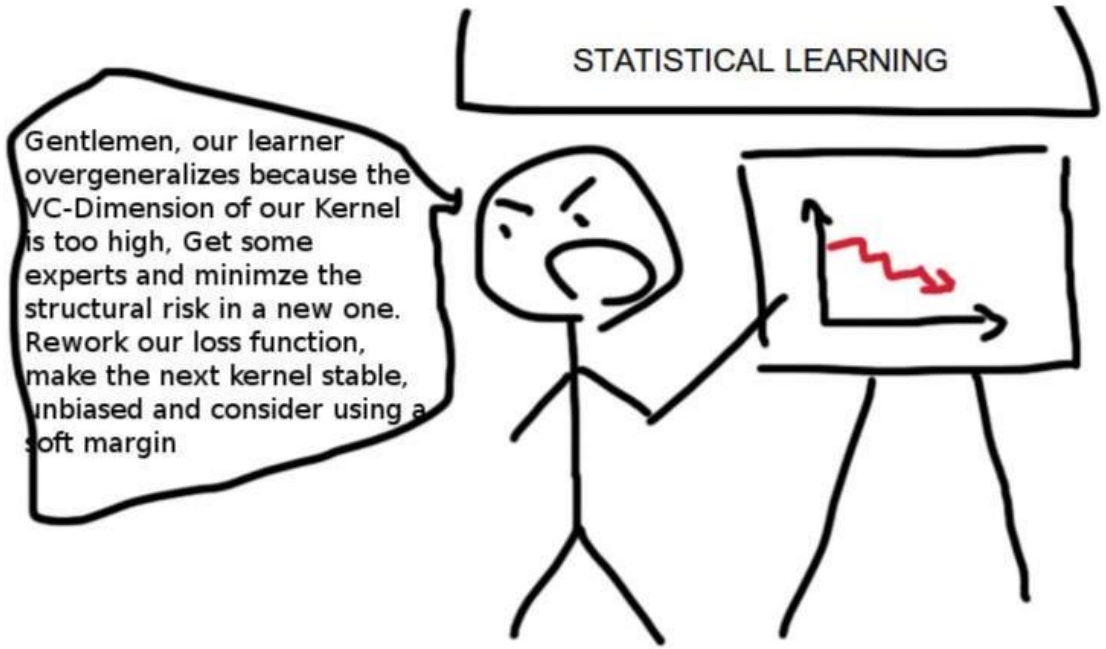
Experiments

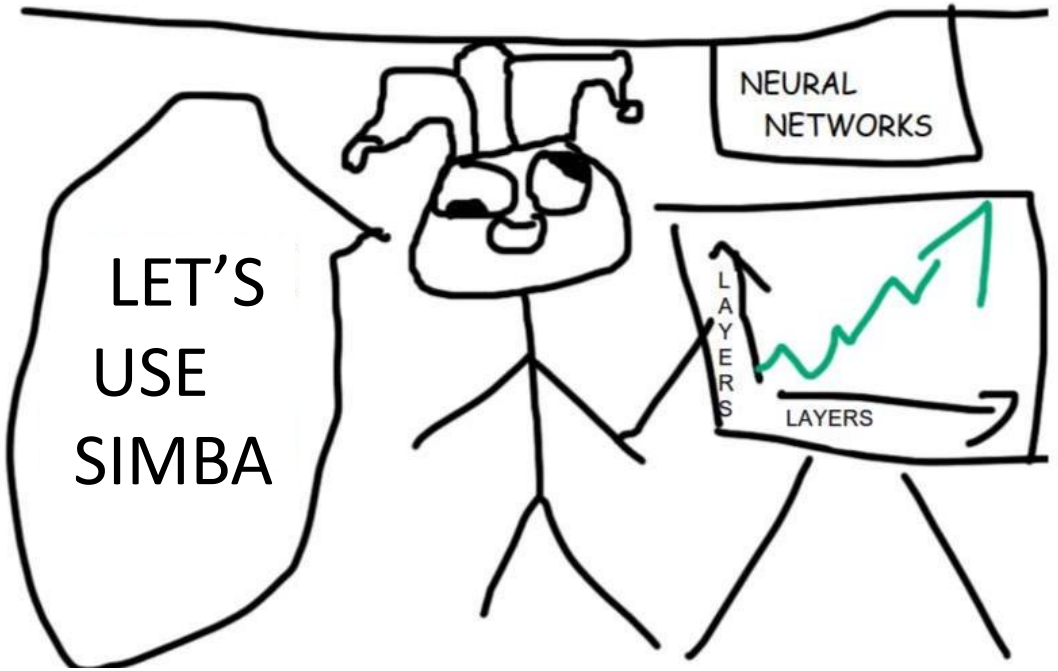
A simple SAC + SimBa combination is highly competitive with SOTA algorithms



*Points in the upper-left corner (↖) indicate higher compute efficiency.

Summary


STATISTICAL LEARNING
Gentlemen, our learner overgeneralizes because the VC-Dimension of our Kernel is too high, Get some experts and minimze the structural risk in a new one. Rework our loss function, make the next kernel stable, unbiased and consider using a soft margin


LET'S USE SIMBA
NEURAL NETWORKS
LAYERS
LAYERS

Why Scaling Laws Do not Hold in Deep RL?

1. Simplicity Bias

In supervised learning, simplicity bias is a key factor driving scaling laws.

Maybe the RL community is **not fully harnessing the simplicity bias.**

2. Non-stationary Optimization

Unlike SL, RL is trained from non-stationary data distributions.

The network can overfit to earlier data and fail to optimize to new data.

Network reinitialization transforms this setup into a stationary optimization.

Why Scaling Laws Do not Hold in Deep RL?

1. Simplicity Bias

In supervised learning, simplicity bias is a key factor driving scaling laws.

Maybe the RL community is not fully harnessing the simplicity bias.

2. Non-stationary Optimization

Unlike SL, RL is trained from non-stationary data distributions.

The network can overfit to earlier data and fail to optimize to new data.

Network reinitialization transforms this setup into a stationary optimization.

Slow and Steady Wins the Race

Maintaining Plasticity with Hare and Tortoise Networks

Hojoon Lee, Hyeonseo Cho, Hyunseung Kim, Donghu Kim, Dugki Min,
Jaegul Choo, Clare Lyle

KAIST AI Graduate School of AI
KONKUK UNIVERSITY
DeepMind

Training Networks from Non-Stationary Data Distribution

Experimental Setup [1]

- **Goal:** Understand the effects of training networks from non-stationary data distributions.
- **Dataset:** CIFAR-10, CIFAR-100, Tiny-ImageNet.
- **Architecture:** ResNet-18, ViT-Tiny, VGG-16.

```
# warm-starting
for epoch in range(epochs):
    train the neural network from the subset (p%) of the noisy (q%) dataset.
# fine-tuning
for epoch in range(epochs):
    train the neural network from the complete noise-free dataset.
evaluate test accuracy.
```

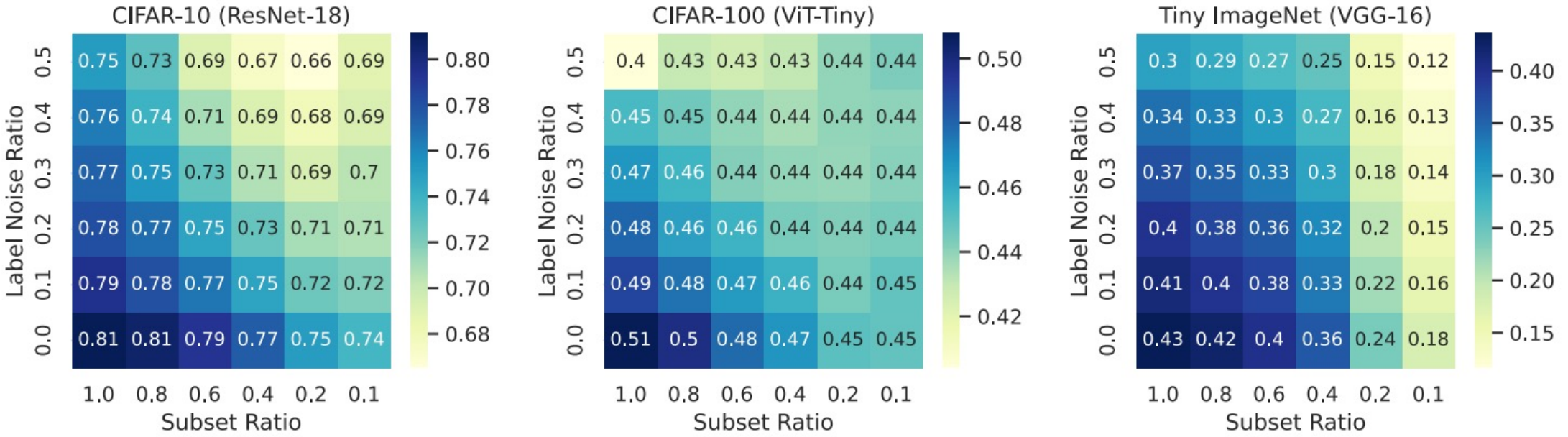
[1] On Warm-Starting Neural Network Training, Ash et al, NeurIPS 2019.

Training Networks from Non-Stationary Data Distribution

Experimental Results

A huge performance drop when trained from smaller subsets and noisy labels.

*a.k.a: loss of plasticity, primacy bias, capacity loss.



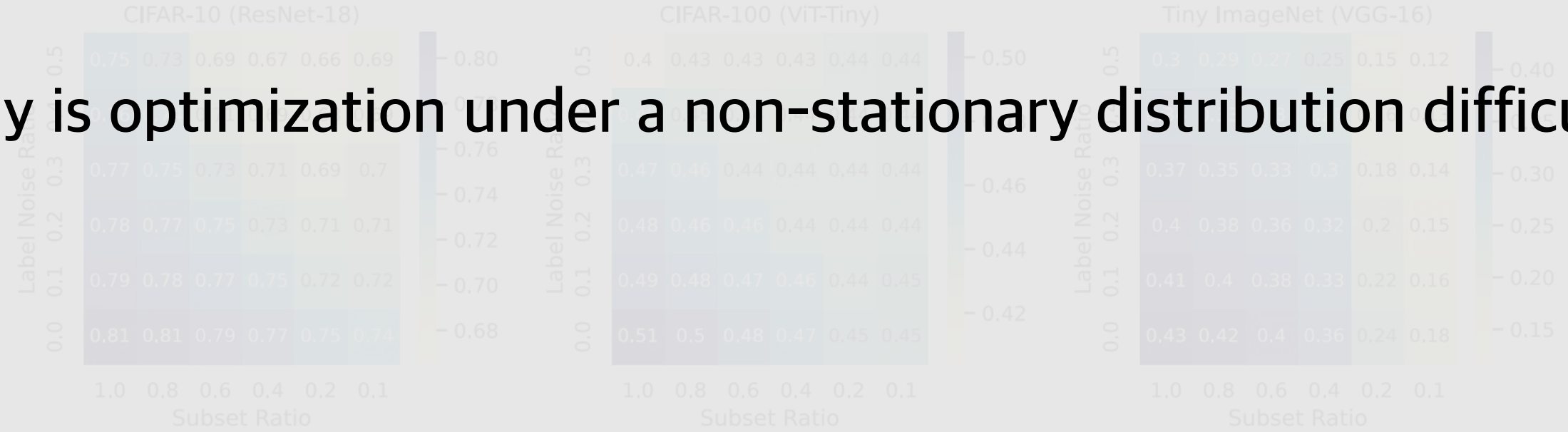
*Points in the upper-right corner (↗) indicate smaller subsets and noisier labels.

Training Networks from Non-Stationary Data Distribution

Experimental Results

A huge performance drop when trained from smaller subsets and noisy labels.

*a.k.a: loss of plasticity, primacy bias, capacity loss.

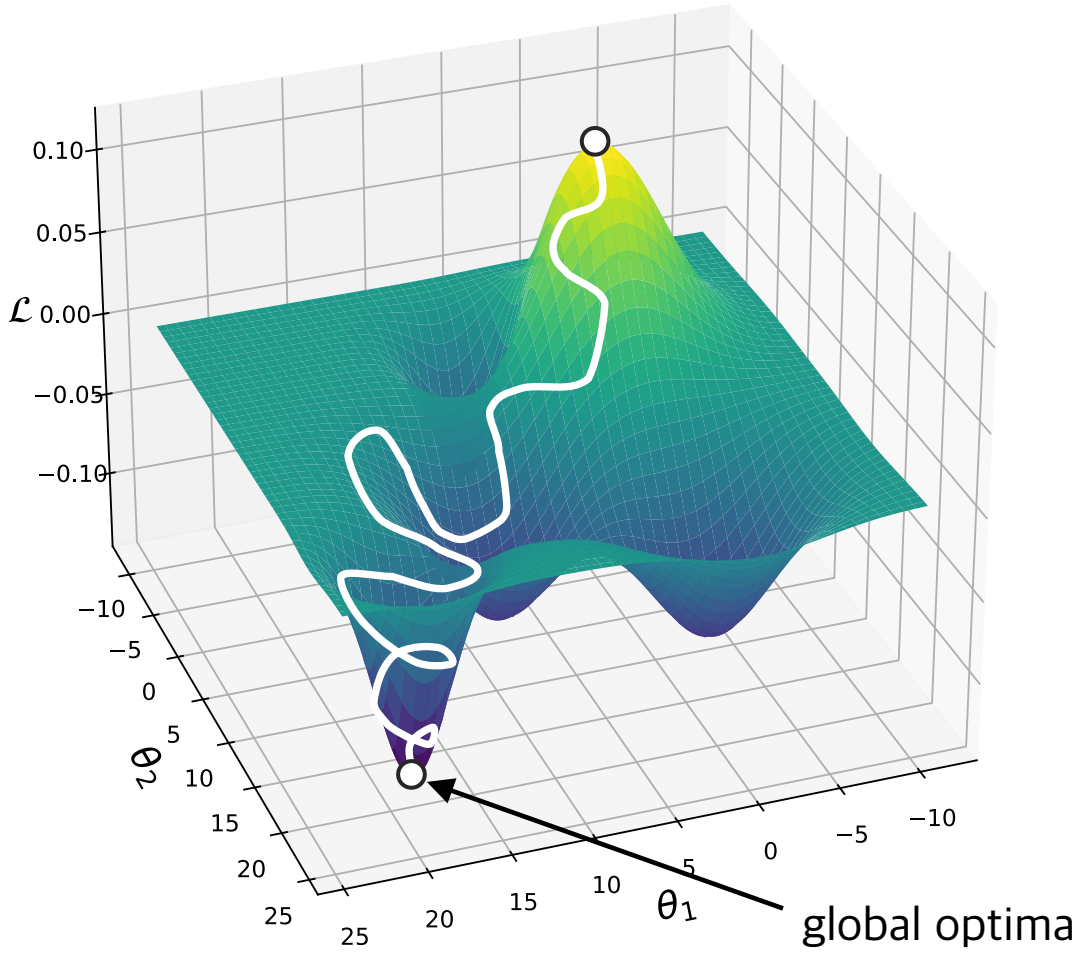


Why is optimization under a non-stationary distribution difficult?

*Points in the upper-right corner (↗) indicate smaller subsets and noisier labels.

Thought Experiment

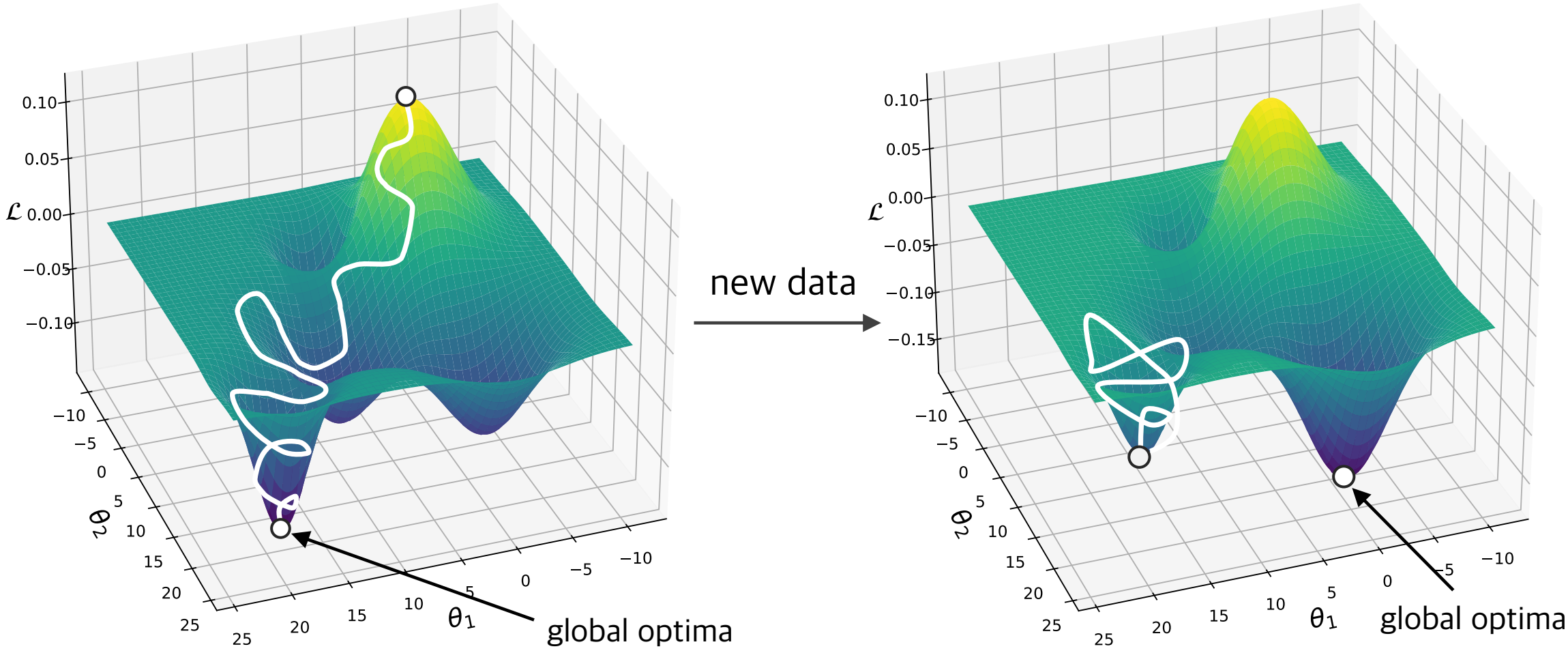
Gradient Descent



Warm-Starting Phase

Thought Experiment

Gradient Descent

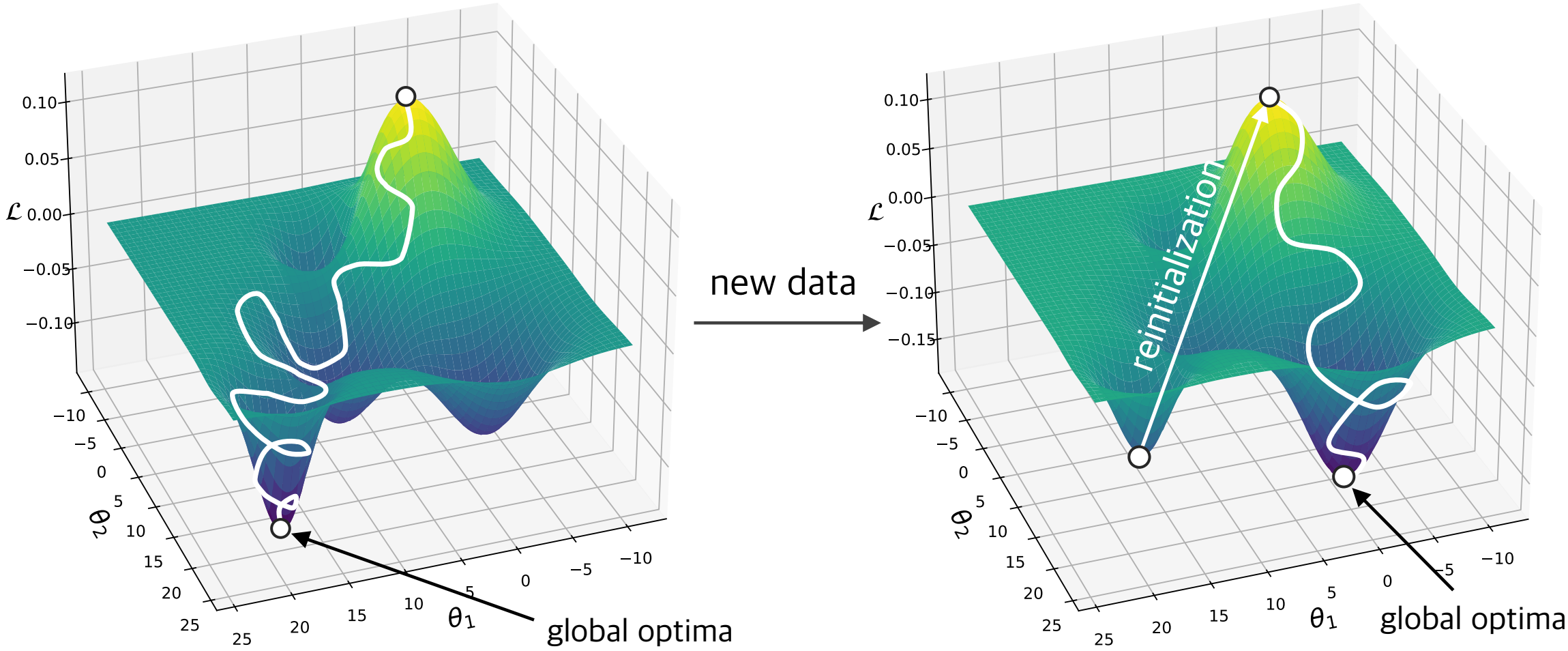


Warm-Starting Phase

Fine-Tuning Phase

Thought Experiment

Gradient Descent + Reinitialization

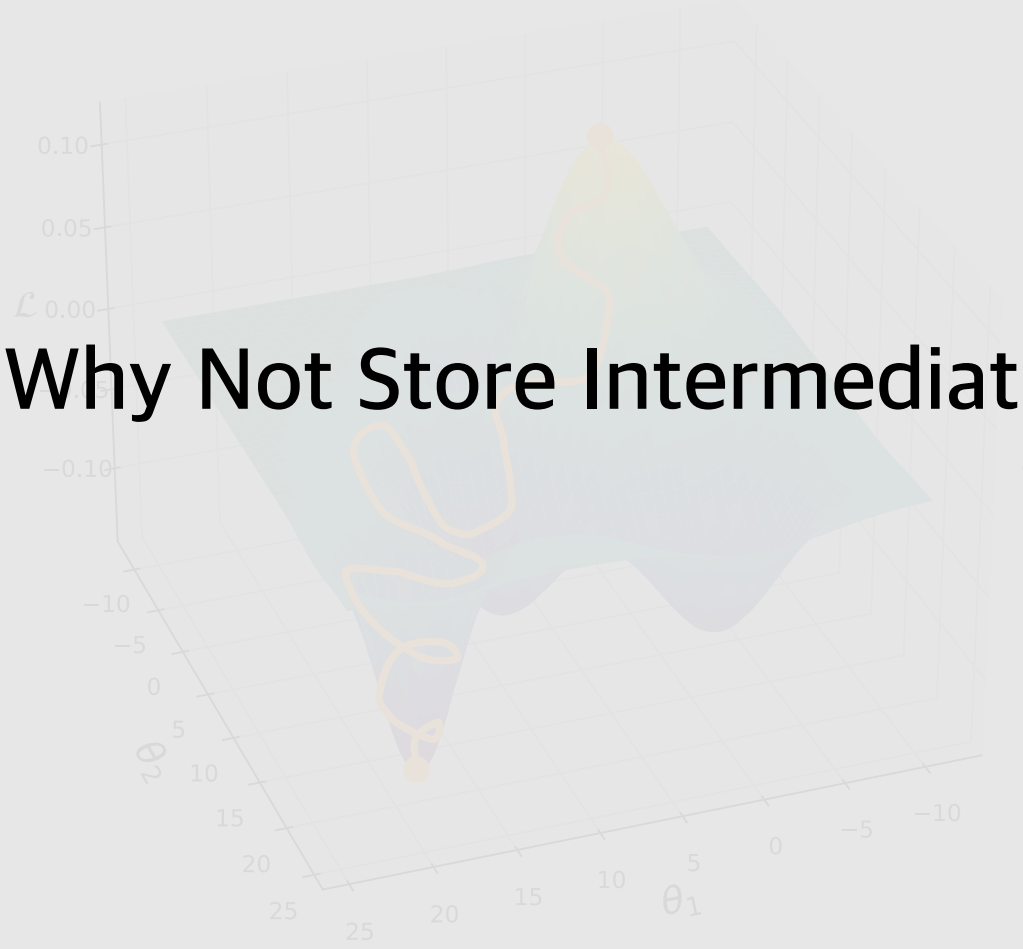


Warm-Starting Phase

Fine-Tuning Phase

Thought Experiment

Gradient Descent + Reinitialization



Warm-Starting Phase

new data

Fine-Tuning Phase

Why Not Store Intermediate Checkpoints for Reinitialization?

Idea: Use EMA Network as a Checkpoint

PseudoCode

for step, (x,y) in enumerate(data_loader):

    # update hare network θ_h

    loss = loss_fn($h(x), y$)

    loss.backward()

    optimizer.step(θ_h)

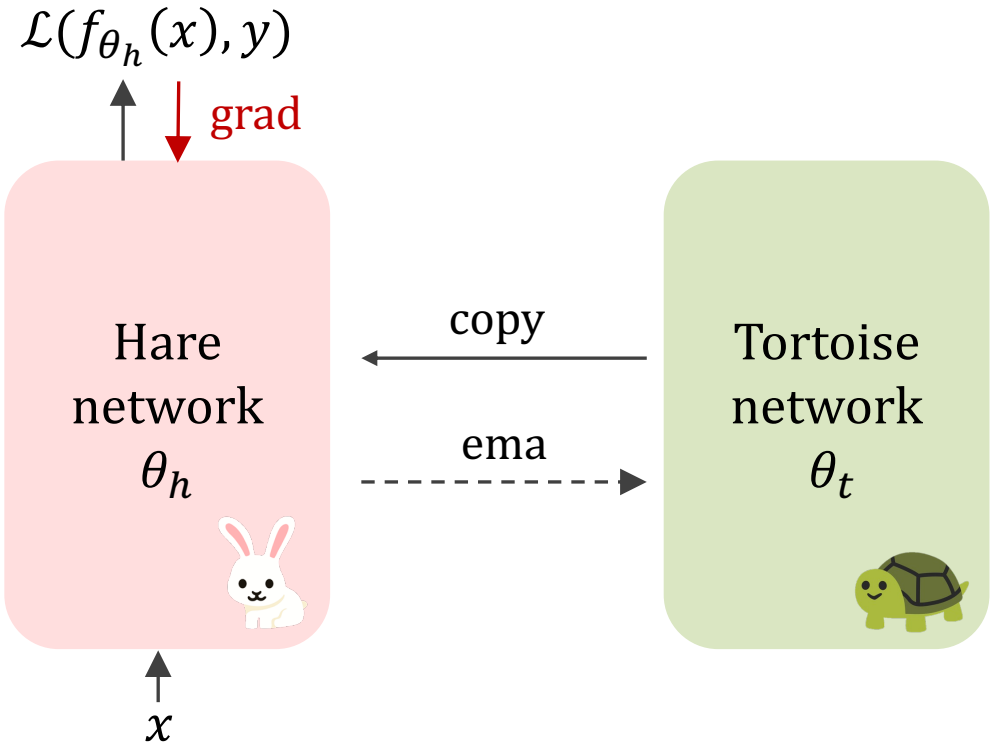
    # update tortoise network θ_t

    $\theta_h = m \times \theta_t + (1 - m) \times \theta_h$

    # reinitialize hare to tortoise

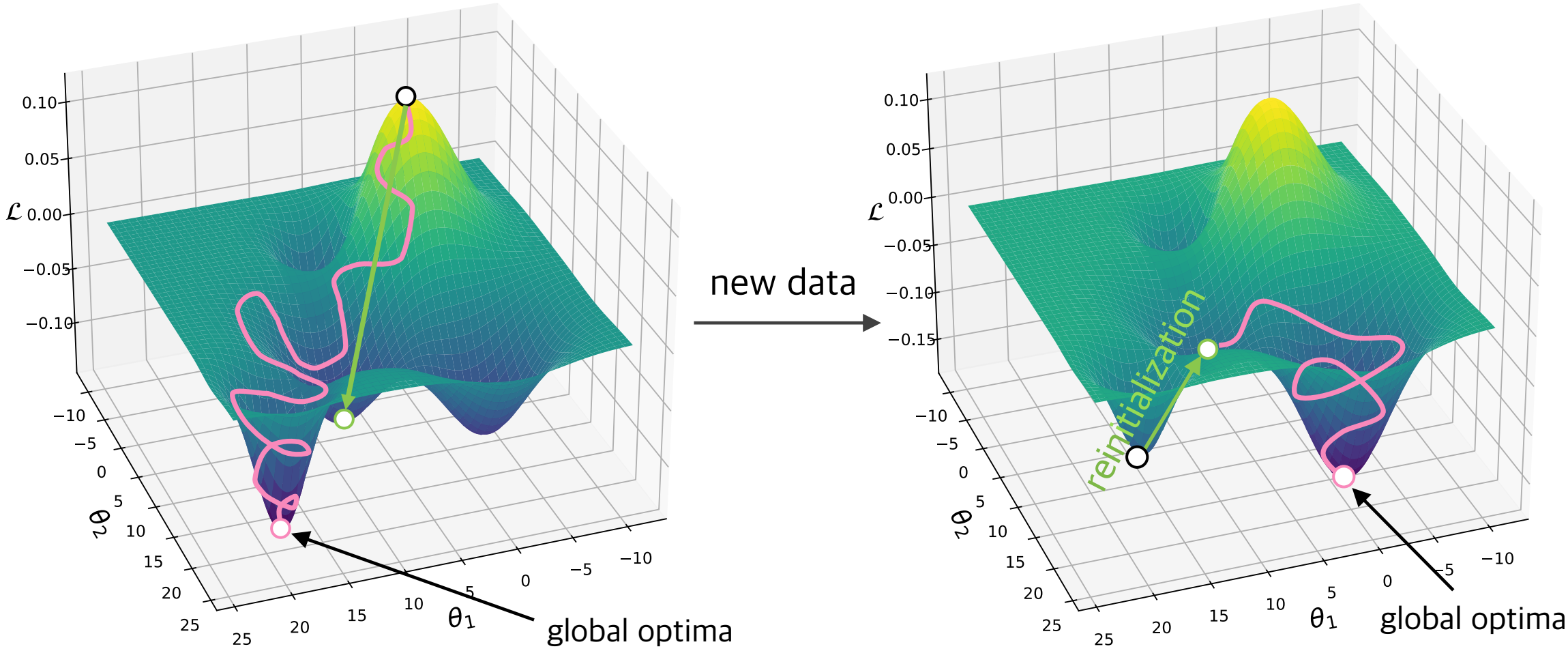
    if step % reinitialization_interval:

        $\theta_h = \theta_t$



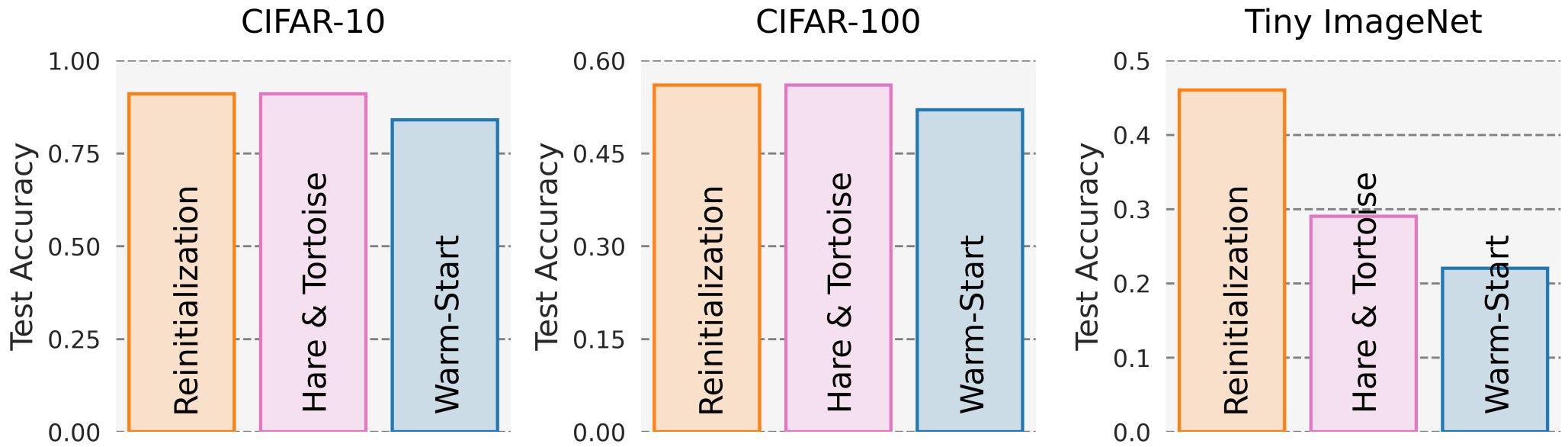
Thought Experiment

Gradient Descent + Hare & Tortoise



Experiments

Hare & Tortoise effectively reduces overfitting in non-stationary optimization scenario.



Why Scaling Laws Do not Hold in Deep RL?

1. Simplicity Bias

In supervised learning, simplicity bias is a key factor driving scaling laws.

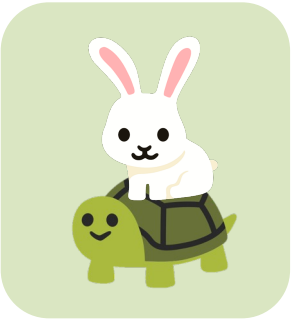
Maybe the RL community is not fully harnessing the simplicity bias.

2. Non-stationary Optimization

Unlike SL, RL is trained from non-stationary data distributions.

The network can overfit to earlier data and fail to optimize to new data.

Network reinitialization transforms this setup into a stationary optimization.



Why Scaling Laws Do not Hold in Deep RL?

1. Simplicity Bias

In supervised learning, simplicity bias is a key factor driving scaling laws.

Maybe the RL community is not fully harnessing the simplicity bias.

2. Non-stationary Optimization

Unlike SL, RL is trained from non-stationary data distributions.

The network can overfit to earlier data and fail to optimize to new data.

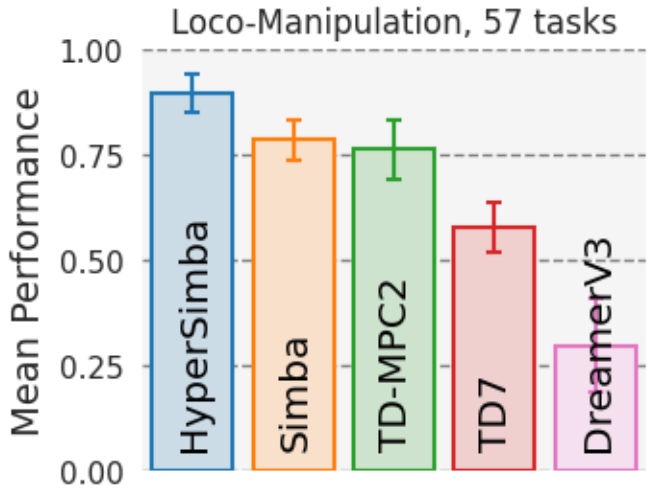
Network reinitialization transforms this setup into a stationary optimization.

Problems Solved?

Future Research Direction

- Leveraging Simplicity Bias to Broader Setup
  - Vision-Based RL.
  - Offline-to-Online RL.
- Measuring the loss of plasticity of neural network
  - Dormant ratio? Parameter Norm? [1,2]
- Designing new Architecture for Non-Stationary Optimization
  - Better update rule? [3]
  - Dynamic neural network? [4]
  - Well normalized architecture? [5]

Teaser



[1] The Dormant Neuron Phenomenon in Deep Reinforcement Learning, Sokar et al, ICML 2023.
[2] Loss of Plasticity in Deep Continual Learning, Dohare et al, Nature 2024.
[3] Addressing Loss of Plasticity and Catastrophic Forgetting in Continual Learning, Elsayed et al, ICLR 2024.
[4] Deep Reinforcement Learning with Plasticity Injection, Nikishin et al, NeurIPS 2024.
[5] nGPT: Normalized Transformer with Representation Learning on the Hypersphere, Loshchilov et al, arXiv 2024.

Advice on Experiments

- Measure everything
  - Always look at qualitative results.
  - Plot correlation charts between metrics.
- Implement from scratch
  - If time permits, try re-implementing algorithms from scratch to understand the details.
- Think big, Start small
  - My experimental growing journey
    - **Bachelor**: Fancy MARL benchmark.
    - **Master**: ImageNet.
    - **Early PhD**: CIFAR-10, Atari.
    - **Mid PhD**: Synthetic Toy-set, CartPole.

https://spinningup.openai.com/en/latest/spinningup/spinningup.html

Thank You for Your Attention!

Hojoon Lee

joonleesky@kaist.ac.kr

joonleesky.github.io

Thanks to my collaborators and advisor as well!

DAVIAN Data and Visual Analytics Lab

KAIST AI Graduate School of AI