A Theory of How Pretraining Shapes Inductive Bias in Fine-Tuning

Dr Clémentine C. J. Dominé
ISTA → Harvard

YIHD 2026, Trieste

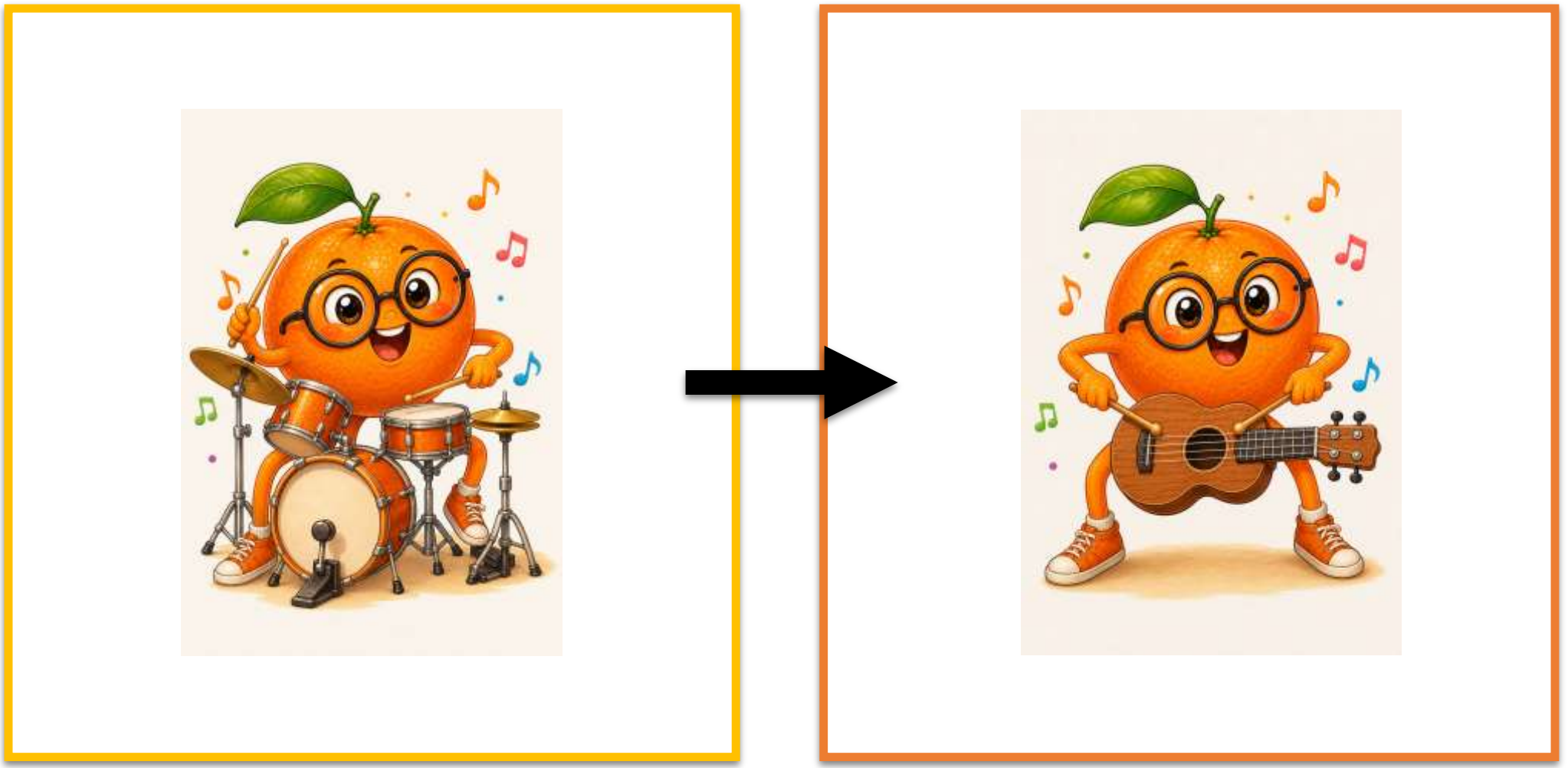
Introduction

Auxiliary tasks play an important role in learning new tasks

Introduction

Auxiliary tasks play an important role in learning new tasks

Introduction



We don't have a good theoretical understanding of how auxiliary tasks help

Introduction

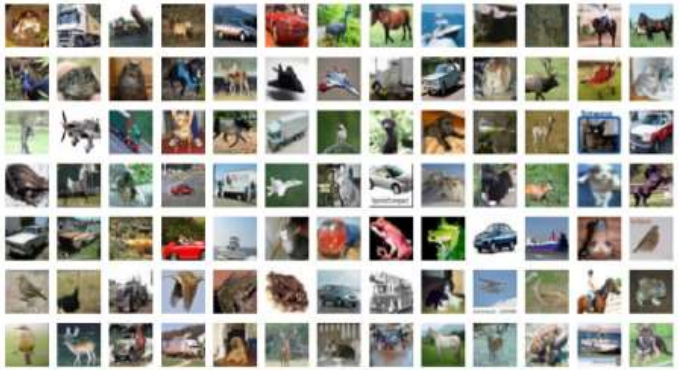
Pretraini
ng

We don't have a good theoretical understanding of how pretraining tasks help

Introduction

Pretraining

Fine-tuning

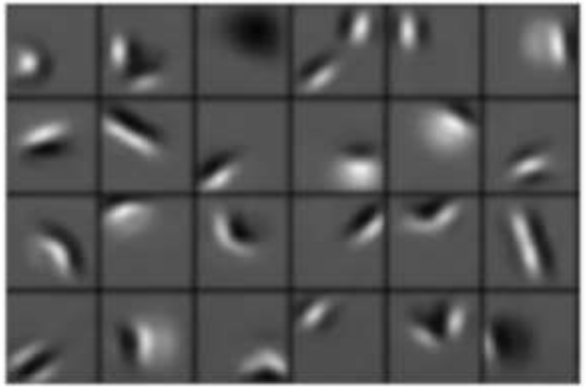


We don't have a good theoretical understanding of how pretraining tasks help

Introduction

Pretraining

Feature learning



Edges, dark spots

Fine-tuning

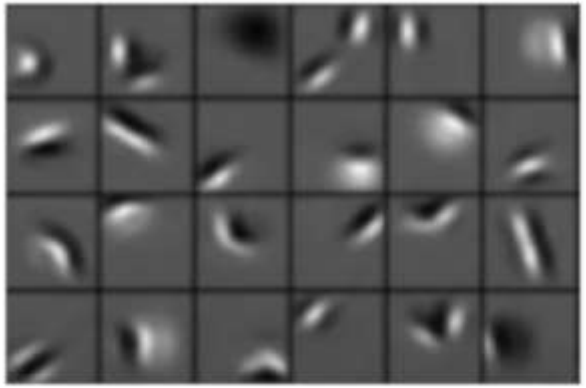
- Feature reuse

We don't have a good theoretical understanding of how pretraining tasks help

Introduction

Pretraining

Feature learning



Edges, dark spots

Fine-tuning

- Feature reuse
- Feature learning

We don't have a good theoretical understanding of how pretraining tasks help

Contribution

How does pretraining shapes the inductive bias of the network during fine-tuning?

A minimal model of the inductive bias of pretraining and fine-tuning with applications to practical settings

Outline

1. **From single-task learning to PT+FT:**
    1. Implicit bias of single task
    2. Implicit bias of pretraining and fine-tuning
2. **From implicit bias to generalization**
3. **Do the same levers matter in practical networks?**
4. **Conclusion**

1. From single-task learning to PT+FT

The starting point is the rich/lazy distinction.

1.1 The implicit bias of pretraining

1.1 Implicit bias of single task

Pretraining

What controls feature learning?

Fine-tuning

1.1 Implicit bias of single task

Pretraining

Learning regimes

Lazy learning / Kernel regime

The network learns a solution similar to kernel regression with a static kernel called the Neural Tangent Kernel (NTK).

Rich / feature learning

The network learns a solution where the NTK is not static.

1.1 Implicit bias of single task

Pretraining

Learning regimes

Lazy learning / Kernel regime

The network learns a solution similar to kernel regression with a static kernel called the Neural Tangent Kernel (NTK).

Rich / feature learning

The network learns a solution where the NTK is not static.

Controlled by

Initialisation absolute scale

Saxe et al., 2019; Chizat et al., 2019; Woodworth et al., 2020.

Output scale

Jacot et al., 2018.

Initialisation relative scale

Xu & Ziyin, 2024; Kunin et al., 2024; Dominé et al., 2025; Jarvis et al., 2025.

1.1 Implicit bias of single task

Pretraining

Implicit bias

These two dynamical regimes lead to different interpolating solutions and implicit bias.

Lazy ↔ Dense bias

Network uses a dense set of features

Rich ↔ Sparse bias

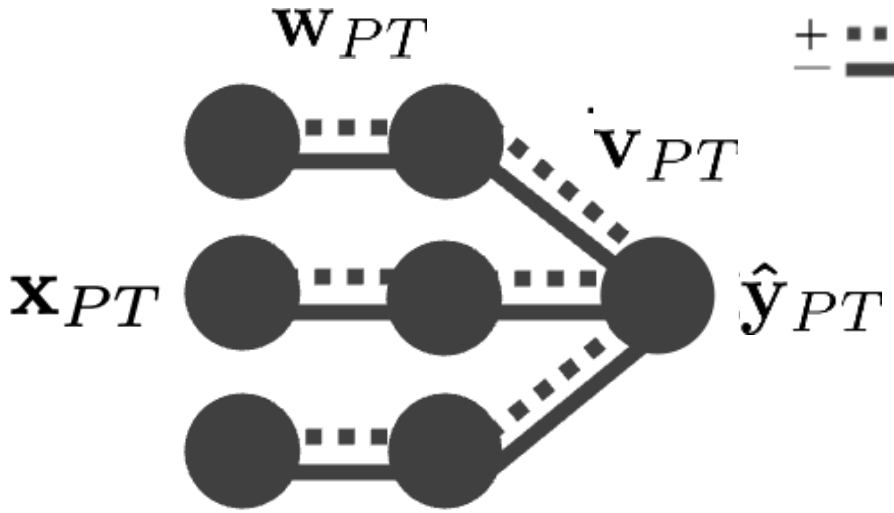
Network uses a sparse set features

[Gunasekar et al., 2018 Woodworth et al., 2020 Azulay et al., 2021]

1.1 Implicit bias of single task

Pretraining

Setup Diagonal linear networks trained with gradient flow until convergence



$$f_{\mathbf{w},\mathbf{v}}(\mathbf{x}) = \beta(\mathbf{w},\mathbf{v})^T\mathbf{x},$$

$$\beta(\mathbf{w},\mathbf{v}) := \mathbf{v}^+ \circ \mathbf{w}^+ - \mathbf{v}^- \circ \mathbf{w}^- \in \mathbb{R}^D$$

Diagonal linear networks: a minimal theoretical model of feature learning

1.1 Implicit bias of single task

Pretraini

Theorem 1.1. *For unbiased initialization, suppose that the gradient flow solution $\beta_{\mathrm{PT}}(\infty)$ satisfies $X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}}(\infty) = y_{\mathrm{PT}}$. Then the gradient flow solution at convergence is given by*

$$\beta_{\mathrm{PT}}(\infty) = \arg\min_{\beta_{\mathrm{PT}}} Q_k(\beta_{\mathrm{PT}}) \quad s.t. \quad X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}} = y_{\mathrm{PT}},$$

where $$Q_k(\beta_{\mathrm{PT}}) = \sum_{d=1}^{D} q_{k_d}(\beta_{\mathrm{PT},d}),$$

and $$q_k(z) = \frac{\sqrt{k}}{4}\left[1 - \sqrt{1 + \frac{4z^2}{k}} + \frac{2z}{\sqrt{k}}\operatorname{arcsinh}\left(\frac{2z}{\sqrt{k}}\right)\right],$$

and

$$\sqrt{k_d} = 2\left(\left(w_d^{+}(0)\right)^2 + \left(v_d^{+}(0)\right)^2\right),$$

[Gunasekar et al., 2018 Woodworth et al., 2020 Azulay et al., 2021]

1.1 Implicit bias of single task

Pretraini

Theorem 1.1. *For unbiased initialization, suppose that the gradient flow solution $\beta_{\mathrm{PT}}(\infty)$ satisfies $X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}}(\infty) = y_{\mathrm{PT}}$. Then the gradient flow solution at convergence is given by*

$$\beta_{\mathrm{PT}}(\infty) = \arg\min_{\beta_{\mathrm{PT}}} Q_k(\beta_{\mathrm{PT}}) \quad s.t. \quad X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}} = y_{\mathrm{PT}},$$

where $$Q_k(\beta_{\mathrm{PT}}) = \sum_{d=1}^{D} q_{k_d}(\beta_{\mathrm{PT},d}),$$

and $$q_k(z) = \frac{\sqrt{k}}{4}\left[1 - \sqrt{1 + \frac{4z^2}{k}} + \frac{2z}{\sqrt{k}}\operatorname{arcsinh}\left(\frac{2z}{\sqrt{k}}\right)\right],$$

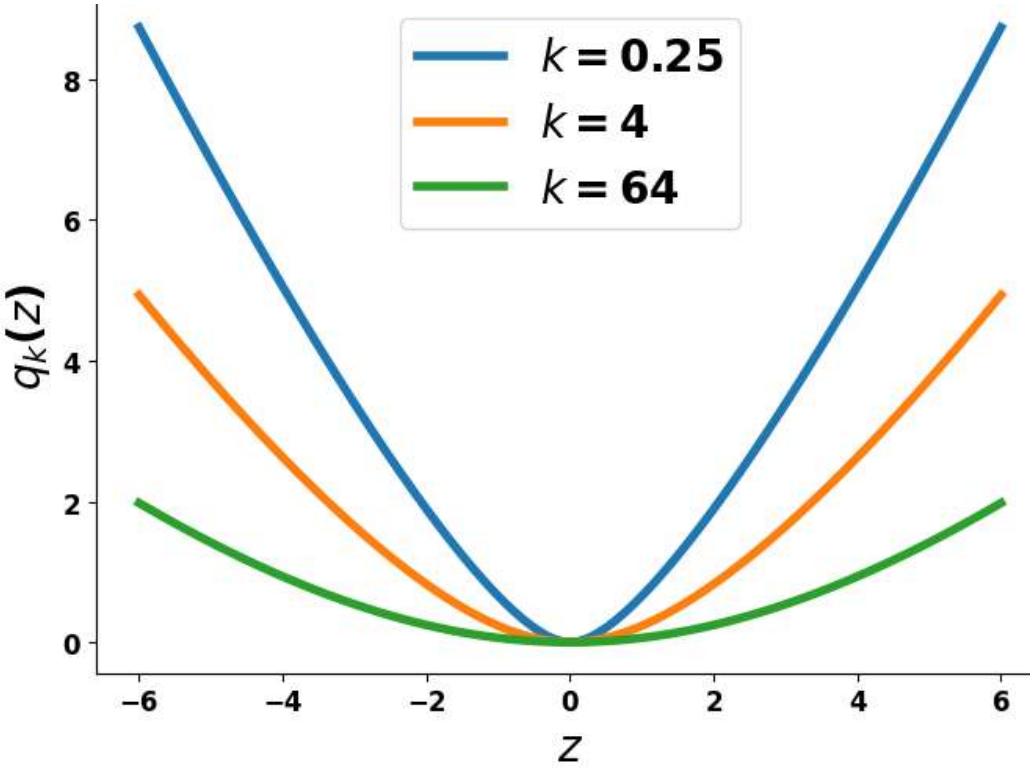
and

$$\sqrt{k_d} = 2\left(\left(w_d^{+}(0)\right)^2 + \left(v_d^{+}(0)\right)^2\right),$$

Initialisation scale

[Gunasekar et al., 2018 Woodworth et al., 2020 Azulay et al., 2021]

Soft inductive bias induced by "Q-penalty"



Q-penalty depends on initial weight magnitude

1.1 Implicit bias of single task

Pretraini

Theorem 1.1. *For unbiased initialization, suppose that the gradient flow solution $\beta_{\mathrm{PT}}(\infty)$ satisfies $X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}}(\infty) = y_{\mathrm{PT}}$. Then the gradient flow solution at convergence is given by*

$$\beta_{\mathrm{PT}}(\infty) = \arg\min_{\beta_{\mathrm{PT}}} Q_k(\beta_{\mathrm{PT}}) \quad s.t. \quad X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}} = y_{\mathrm{PT}},$$

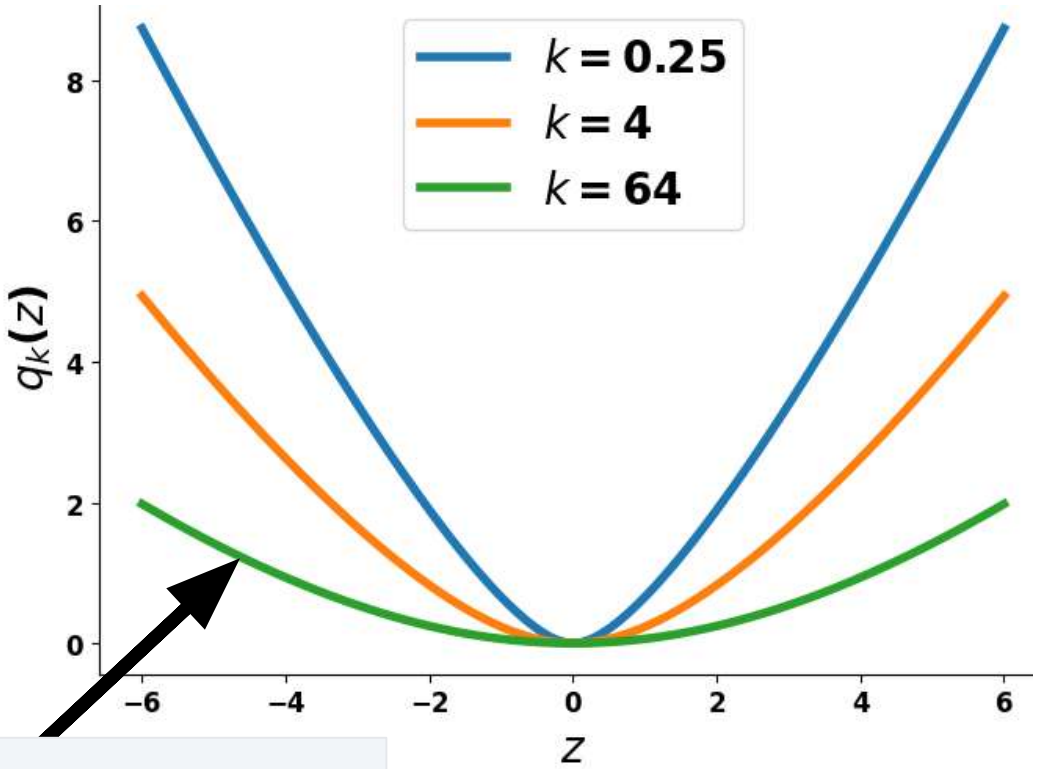
where $$Q_k(\beta_{\mathrm{PT}}) = \sum_{d=1}^{D} q_{k_d}(\beta_{\mathrm{PT},d}),$$

and $$q_k(z) = \frac{\sqrt{k}}{4}\left[1 - \sqrt{1 + \frac{4z^2}{k}} + \frac{2z}{\sqrt{k}}\operatorname{arcsinh}\left(\frac{2z}{\sqrt{k}}\right)\right],$$

and

$$\sqrt{k_d} = 2\left(\left(w_d^{+}(0)\right)^2 + \left(v_d^{+}(0)\right)^2\right),$$

Initialisation scale



Approach to L2 for large k

[Gunasekar et al., 2018 Woodworth et al., 2020 Azu… et al., 2021]

1.1 Implicit bias of single task

Pretraini

Theorem 1.1. *For unbiased initialization, suppose that the gradient flow solution $\beta_{\mathrm{PT}}(\infty)$ satisfies $X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}}(\infty) = y_{\mathrm{PT}}$. Then the gradient flow solution at convergence is given by*

$$\beta_{\mathrm{PT}}(\infty) = \arg\min_{\beta_{\mathrm{PT}}} Q_k(\beta_{\mathrm{PT}}) \quad s.t. \quad X_{\mathrm{PT}}^{\top}\beta_{\mathrm{PT}} = y_{\mathrm{PT}},$$

where $$Q_k(\beta_{\mathrm{PT}}) = \sum_{d=1}^{D} q_{k_d}(\beta_{\mathrm{PT},d}),$$

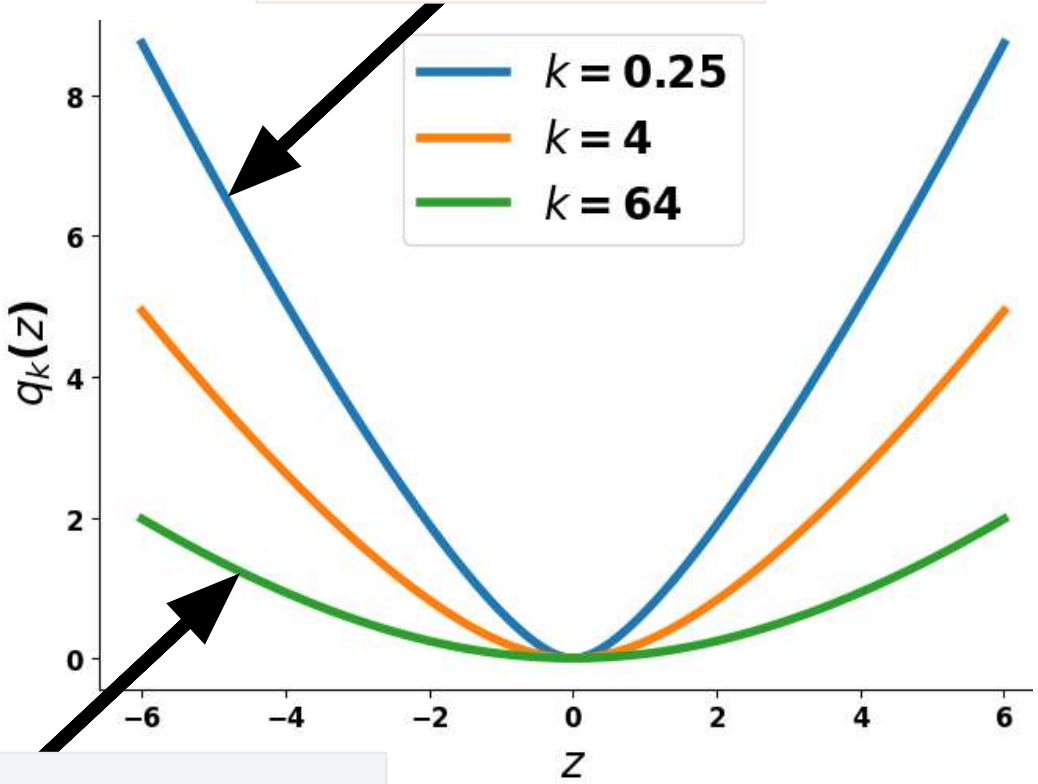
and $$q_k(z) = \frac{\sqrt{k}}{4}\left[1 - \sqrt{1 + \frac{4z^2}{k}} + \frac{2z}{\sqrt{k}}\operatorname{arcsinh}\left(\frac{2z}{\sqrt{k}}\right)\right],$$

and

$$\sqrt{k_d} = 2\left(\left(w_d^{+}(0)\right)^2 + \left(v_d^{+}(0)\right)^2\right),$$

Initialisation scale

Approach to L1 for small k



Approach to L2 for large k

[Gunasekar et al., 2018 Woodworth et al., 2020 Azu
et al., 2021]

1.2 The implicit bias of Pretraining and Fine-tuning

1.2 The implicit bias of PT+FT

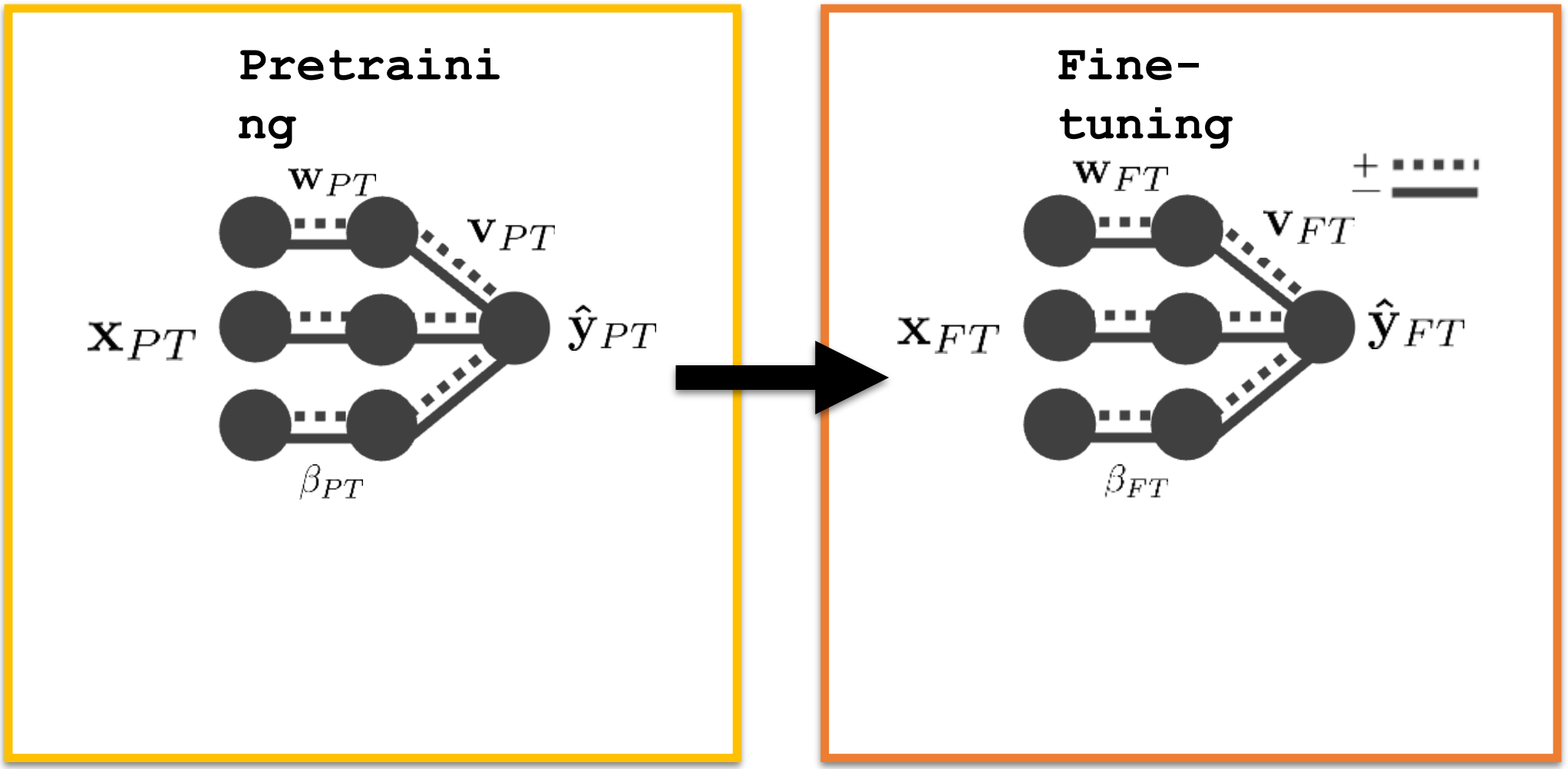
Pretraining

Initialisation controls feature learning

Fine-tuning

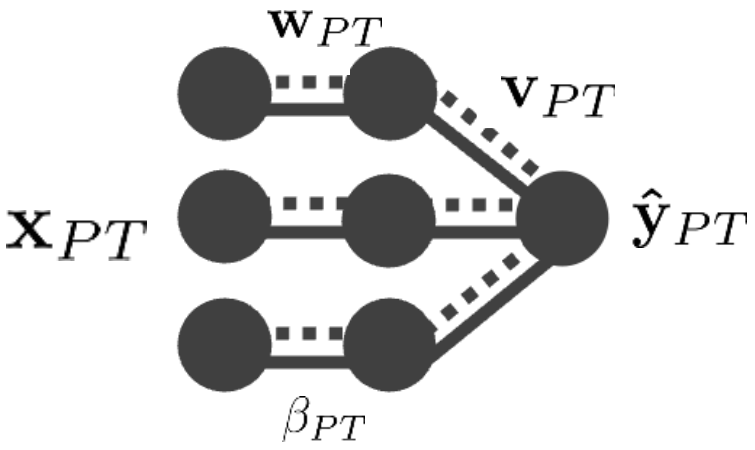
How does pretraining influence the downstream fine-tuning implicit bias ?

1.2 The implicit bias of PT+FT



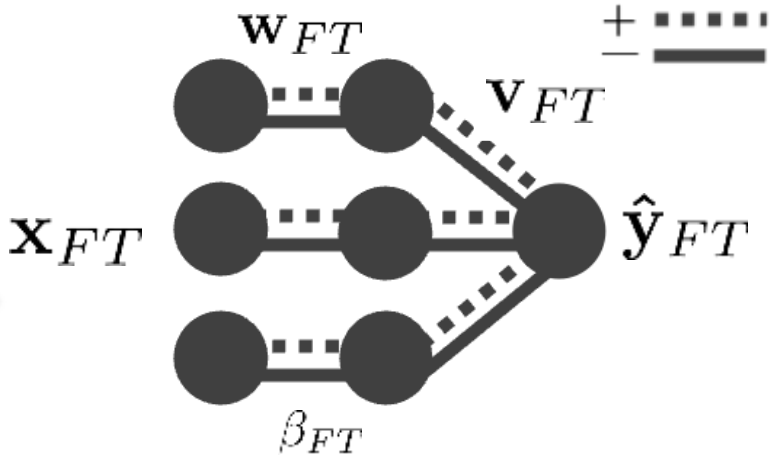
1.2 The implicit bias of PT+FT

Pretraining



$$\lambda_{PT} = (\mathbf{w}_{PT}^{\pm}(0)^2 - \mathbf{v}_{PT}^{\pm}(0)^2)/c_{PT}$$

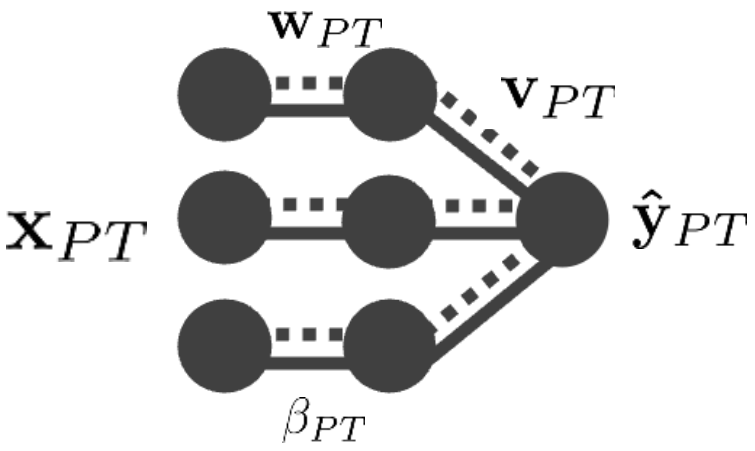
Fine-tuning



Initialisation parameters

1.2 The implicit bias of PT+FT

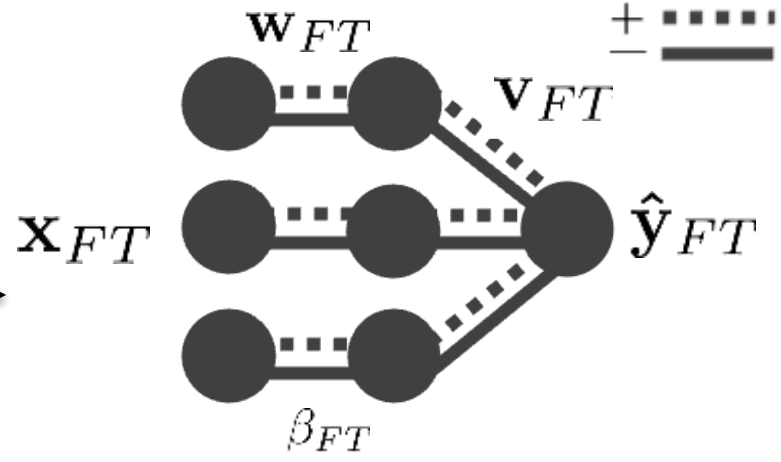
Pretraining



$$\lambda_{PT} = (\mathbf{w}_{PT}^{\pm}(0)^2 - \mathbf{v}_{PT}^{\pm}(0)^2)/c_{PT}$$

$$c_{PT} = (\mathbf{w}_{PT}^{\pm}(0)^2 + \mathbf{v}_{PT}^{\pm}(0)^2)$$

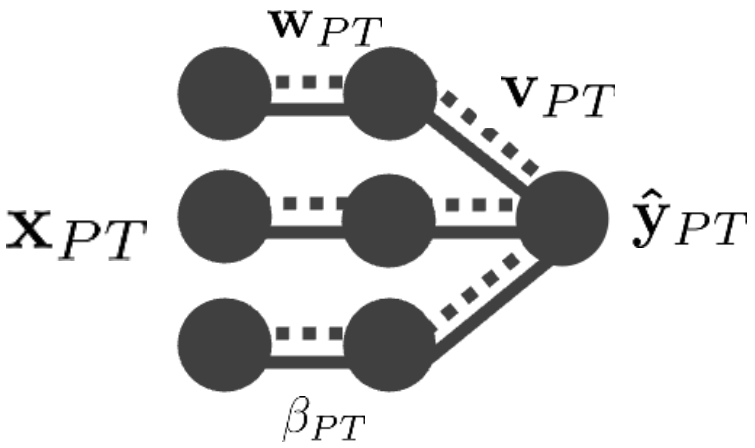
Fine-tuning



Initialisation parameters

1.2 The implicit bias of PT+FT

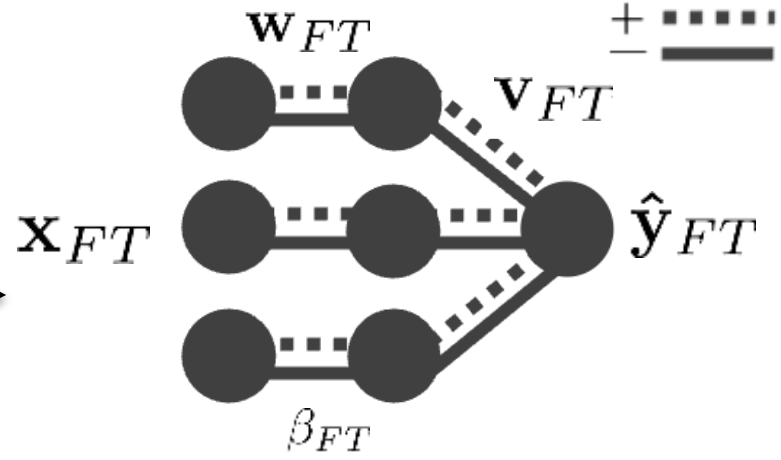
Pretraining



$$\lambda_{PT} = (\mathbf{w}_{PT}^{\pm}(0)^2 - \mathbf{v}_{PT}^{\pm}(0)^2)/c_{PT}$$

$$c_{PT} = (\mathbf{w}_{PT}^{\pm}(0)^2 + \mathbf{v}_{PT}^{\pm}(0)^2)$$

Fine-tuning

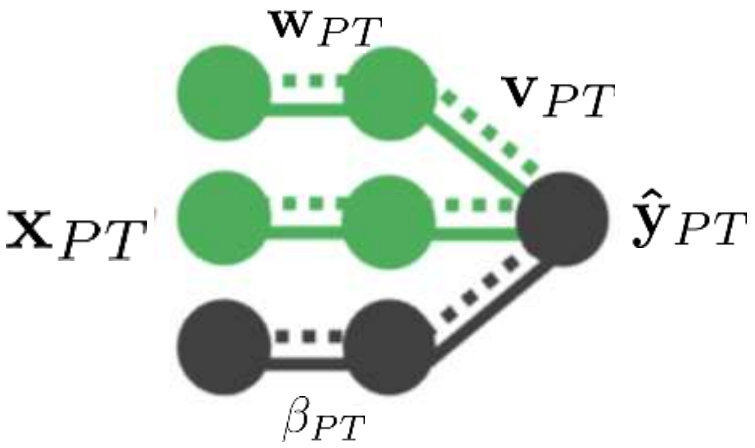


$$\gamma_{FT} = \mathbf{v}_{FT}^{\pm}(0)$$

Initialisation parameters

1.2 The implicit bias of PT+FT

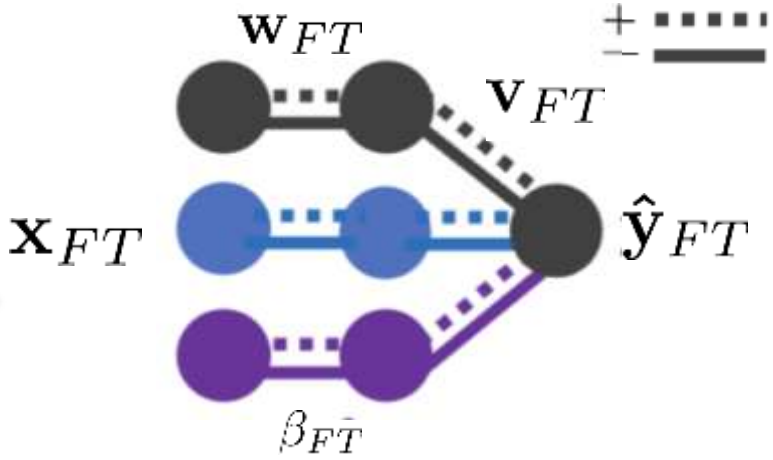
Pretraini
ng



Inactive dimension

ρ_{PT} **Fraction pretrain active dimension**

Fine-tuning



ρ_{FT}^{shared} **Fraction shared active dimension**

ρ_{FT}^{new} **Fraction new active dimensions**

Task Parameters

1.2 The implicit bias of PT+FT

Fine-tuning

Theorem 1.2 (Implicit bias). *Consider a diagonal linear network trained on PT+FT. Then, the gradient flow solution at convergence is given by*

$$\arg\min_{\beta_{FT}} Q_k(\beta_{FT}) \quad s.t.\ \mathbf{X}_{FT}^\top \beta_{FT} = \mathbf{y}_{FT},$$

$$\text{where} \quad Q_k(\beta_{FT}) = \sum_{d=1}^{D} q_{k_d}(\beta_{FT,d}),$$

$$q_k(z) = \frac{\sqrt{k}}{4}\left(1 - \sqrt{1 + \frac{4z^2}{k}} + \frac{2z}{\sqrt{k}} \operatorname{arcsinh}\left(\frac{2z}{\sqrt{k}}\right)\right),$$

$$k_d = \left(2c_{PT}(1+\lambda_{PT})\left(1 + \sqrt{1 + (\hat{\beta}_{PT,d}/c_{PT})^2}\right) + \gamma^2\right)^2.$$

- c_{PT} : Pretraining absolute scale
- λ_{PT} : Pretraining relative scale
- γ_{FT} : Fine-tuning output rescaling
- β_{PT} : Pretraining network function

[See also Lippl & Lindsey (2024)]

1.2 The implicit bias of PT+FT

Fine-tuning

Proposition 1 (Definitions of ℓ-order and Pretraining Dependence). *The implicit bias of fine-tuning is characterized by:*

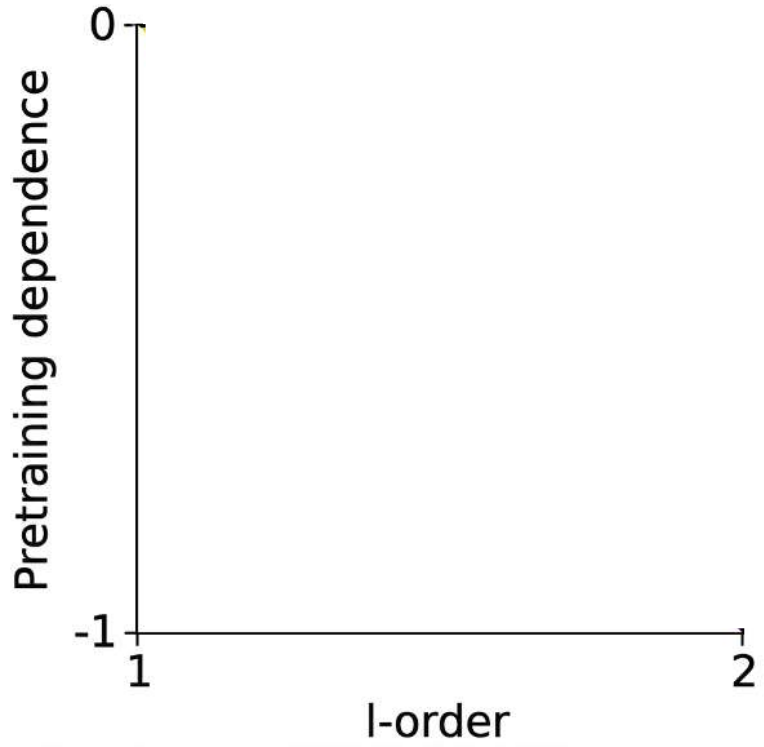
$$\ell\text{-order} := \frac{\partial \log q_{k_d}(\beta_{FT,d})}{\partial \log |\beta_{FT,d}|} \qquad \begin{cases} \approx 1 & \Rightarrow \textit{sparse / rich learning,} \\ \approx 2 & \Rightarrow \ell_2\textit{-like / lazy learning.} \end{cases}$$

$$\mathrm{PD} := \frac{\partial \log q_{k_d}(\beta_{FT,d})}{\partial \log |\beta_{PT,d}|} \qquad \begin{cases} \approx 0 & \Rightarrow \textit{independent of PT features,} \\ < 0 & \Rightarrow \textit{large PT features receive smaller penalties.} \end{cases}$$

[Lippl & Lindsey (2024)]

1.2 The implicit bias of PT+FT

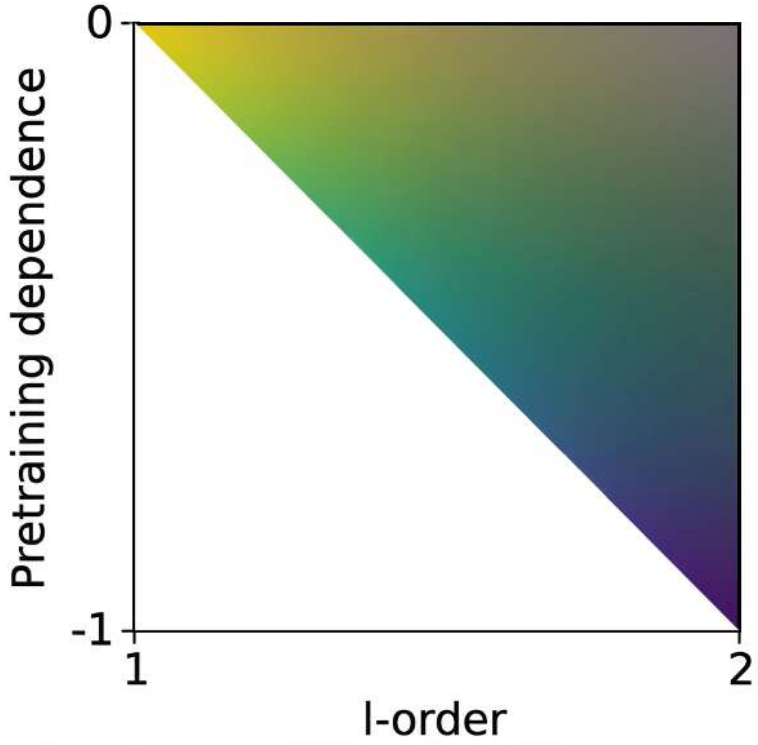
Fine-tuning



We identify four fine-tuning learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning



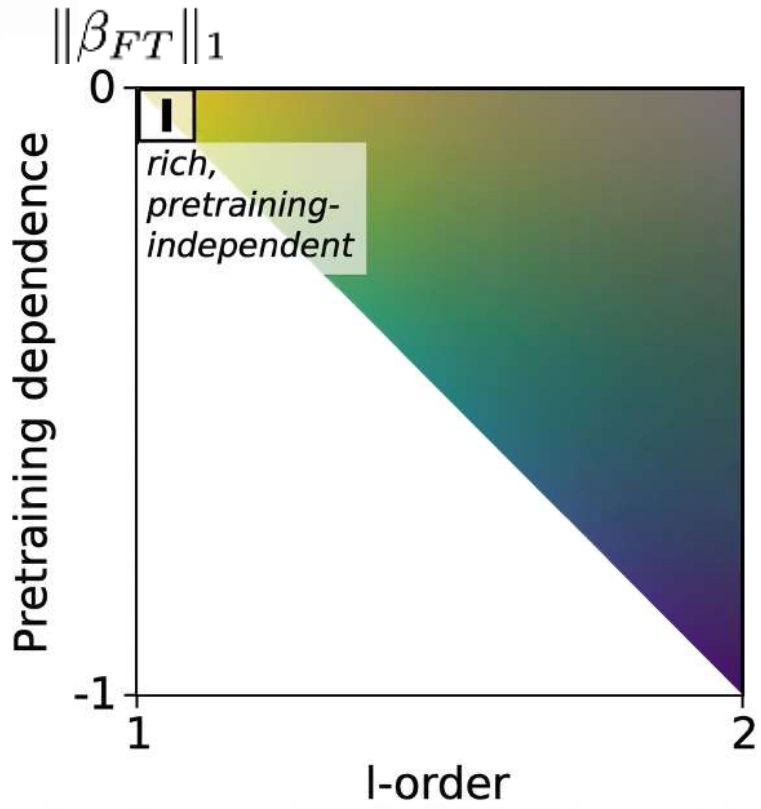
$$\ell\text{-order} \in [1, 2],\ \text{PD} \in [-1, 0],$$

$$\ell\text{-order} + \text{PD} \in [1, 2]$$

We identify four fine-tuning learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning



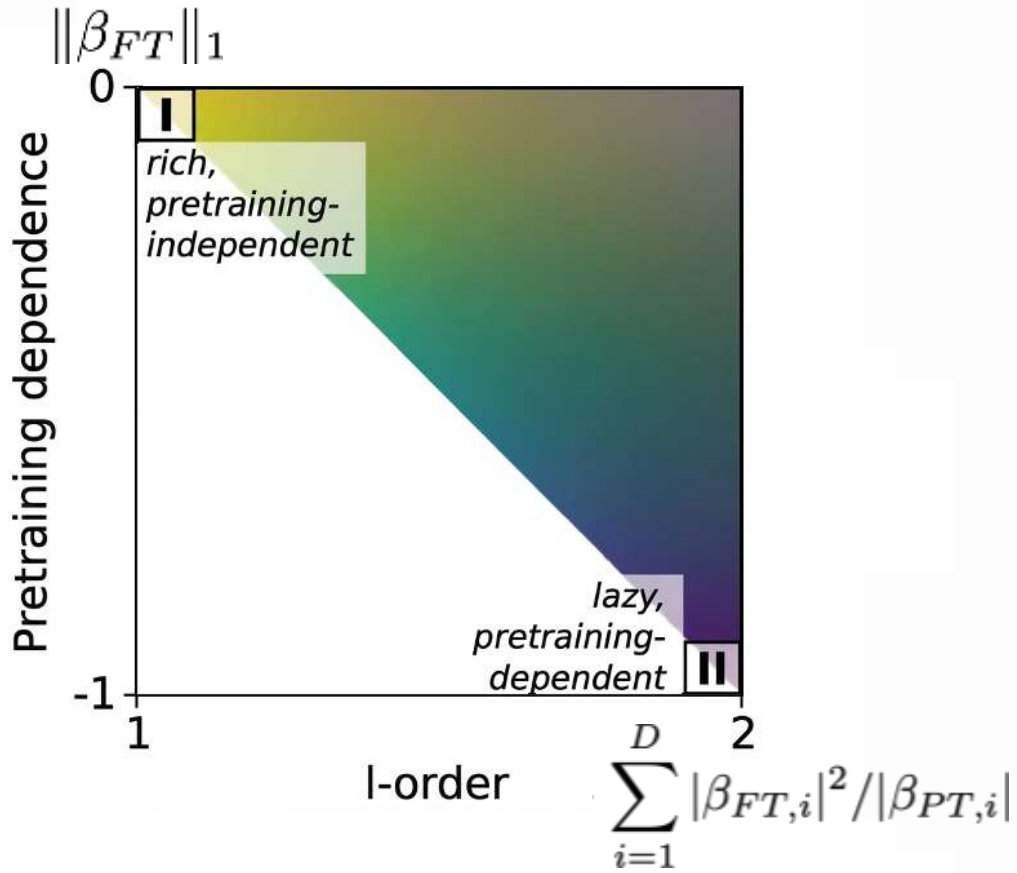
This regime is governed by

$$Q(\beta_{FT}) \to \|\beta_{FT}\|_1$$

We identify four fine-tuning learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning



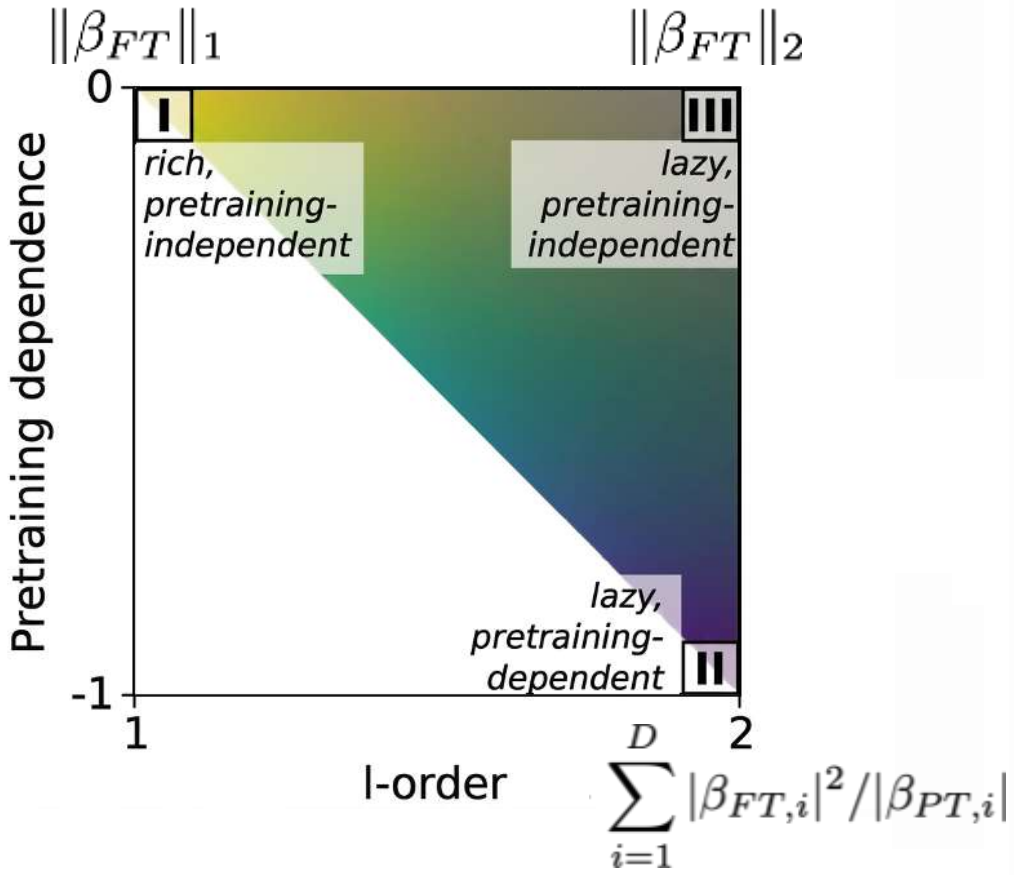
This regime is governed by

$$Q(\beta_{FT}) \to \sum_{i=1}^{D} |\beta_{FT,i}|^2/|\beta_{PT,i}|$$

We identify four fine-tuning learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning



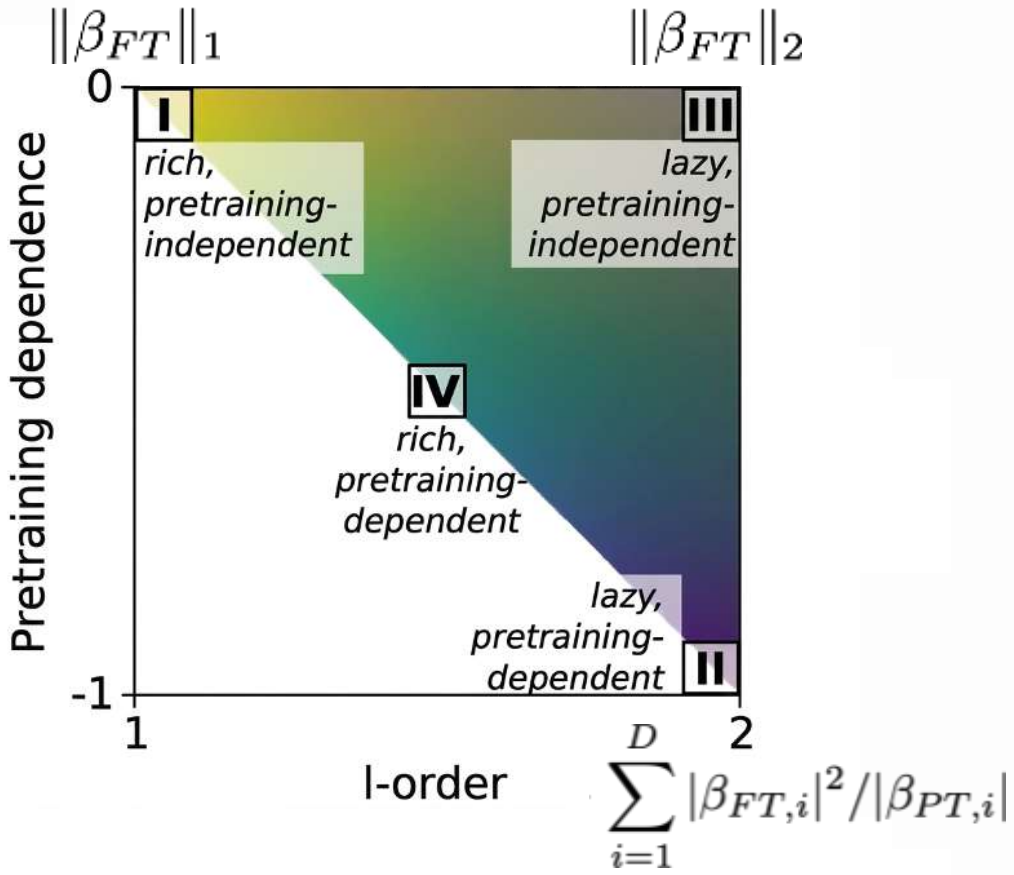
This regime is governed by

$$Q(\beta_{FT}) \to \|\beta_{FT}\|_2$$

We identify four fine-tuning learning regimes

1.2 The implicit bias of PT+FT

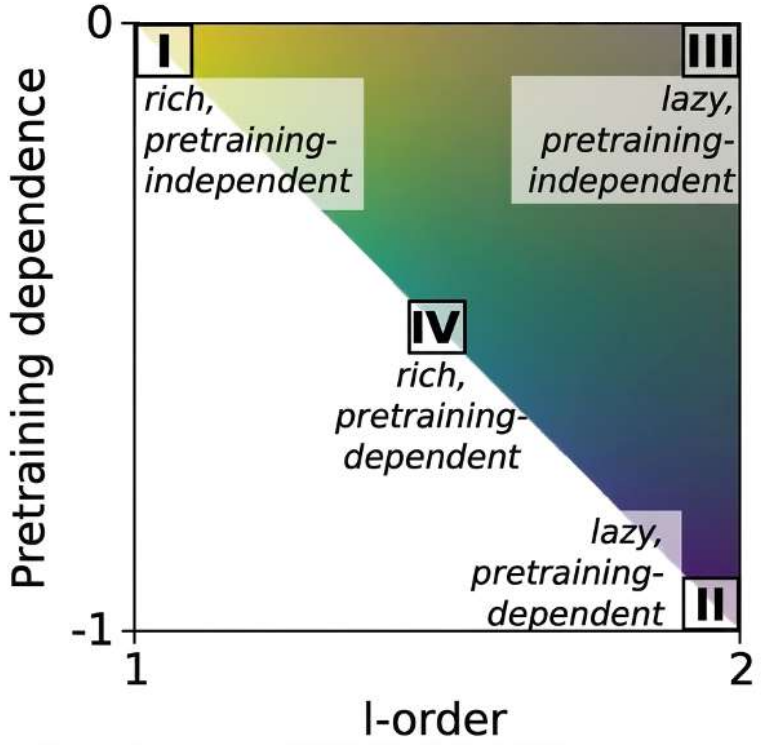
Fine-tuning

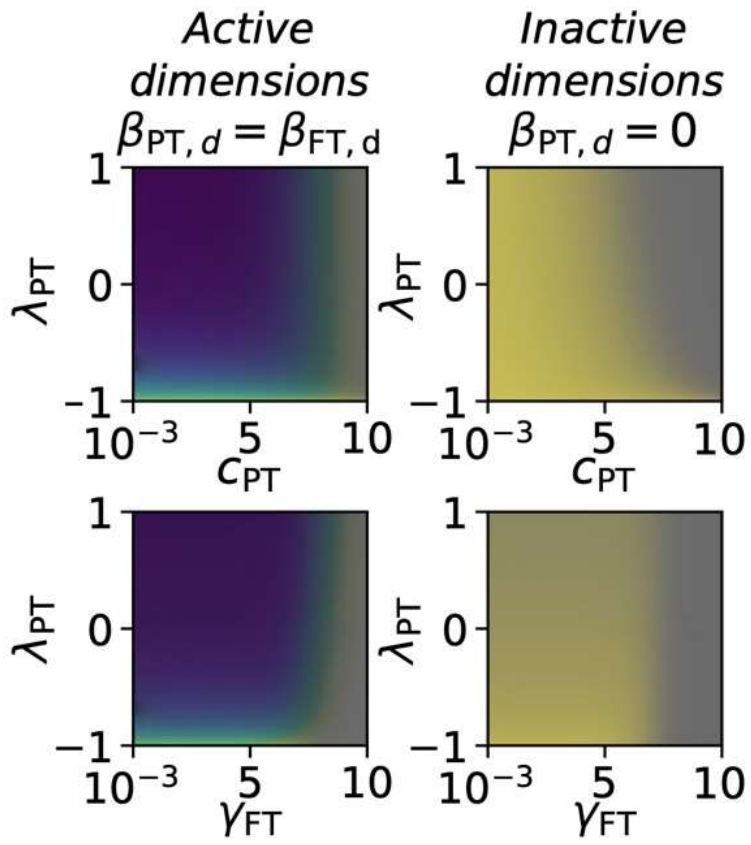


We identify four fine-tuning learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning

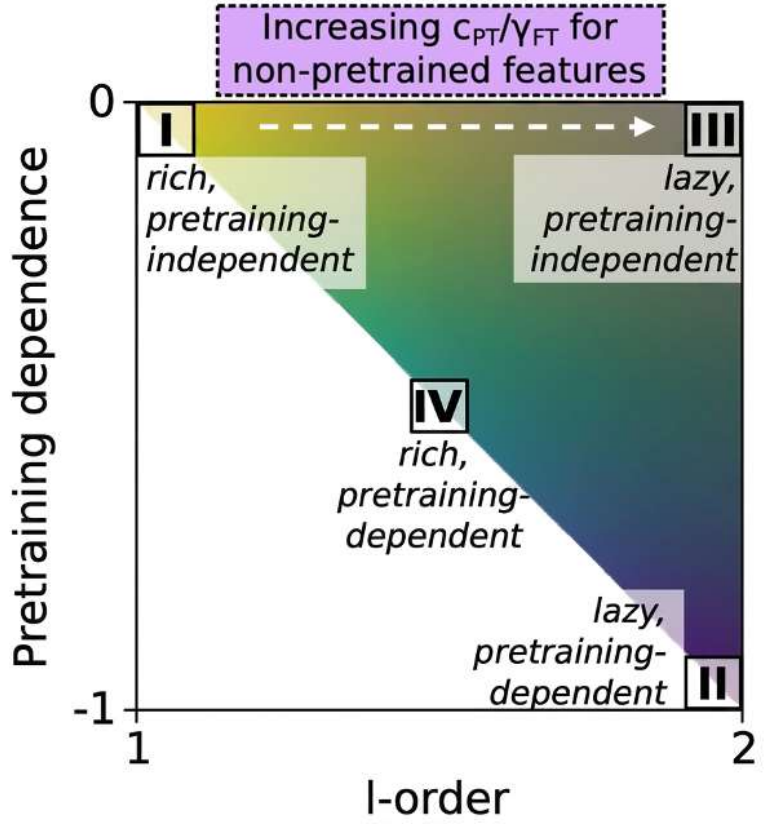


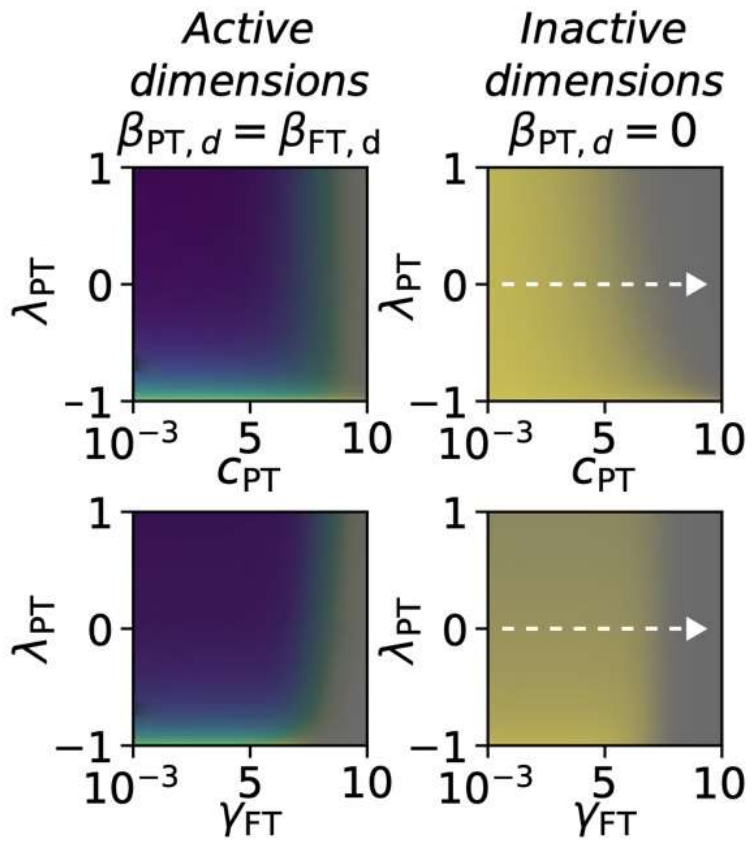


We fully characterise the **implicit bias** as a function of **relative scaling, scale and output rescaling** showcasing different learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning

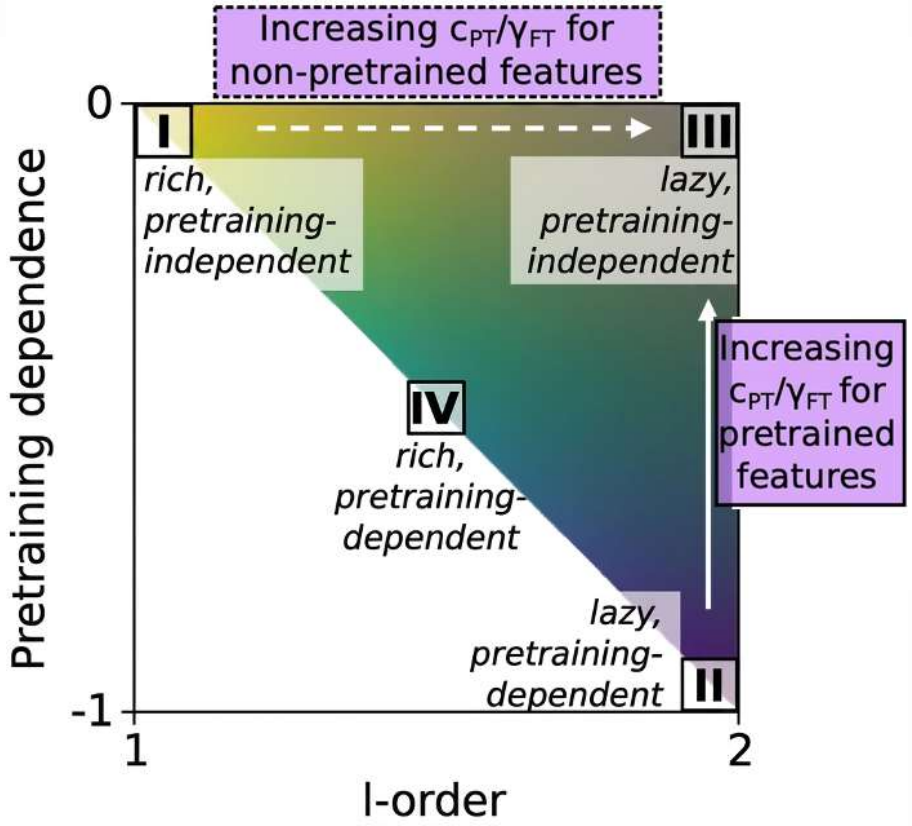


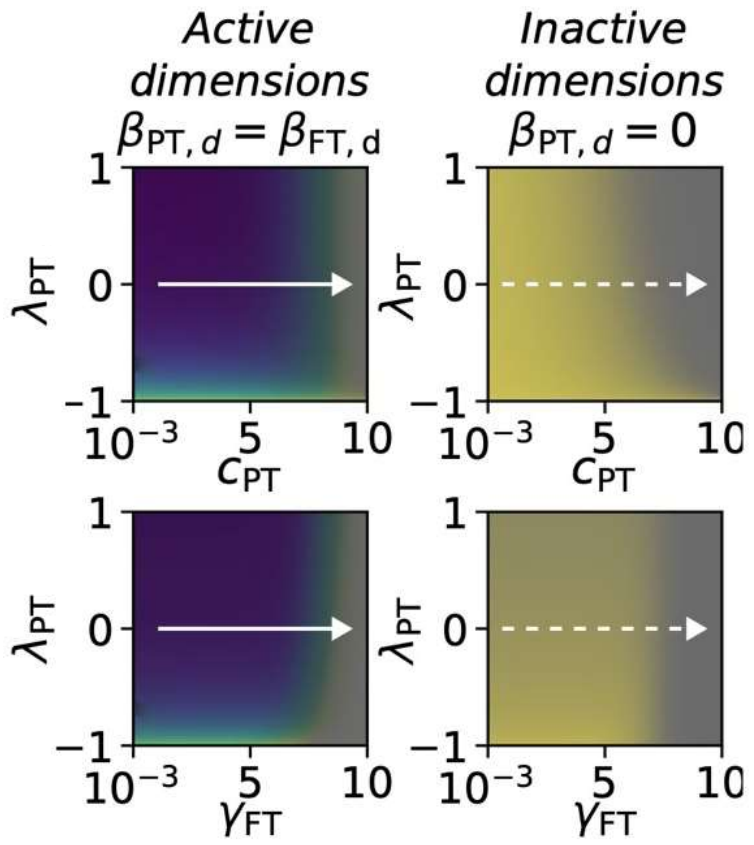


We fully characterise the **implicit bias** as a function of **relative scaling, scale and output rescaling** showcasing different learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning

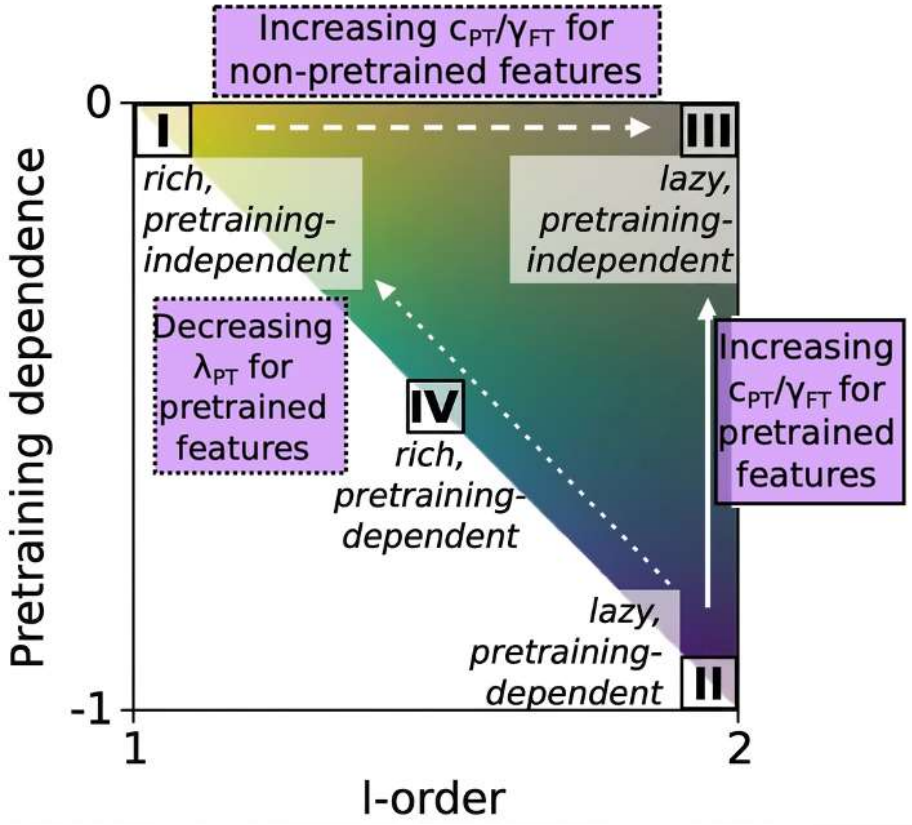


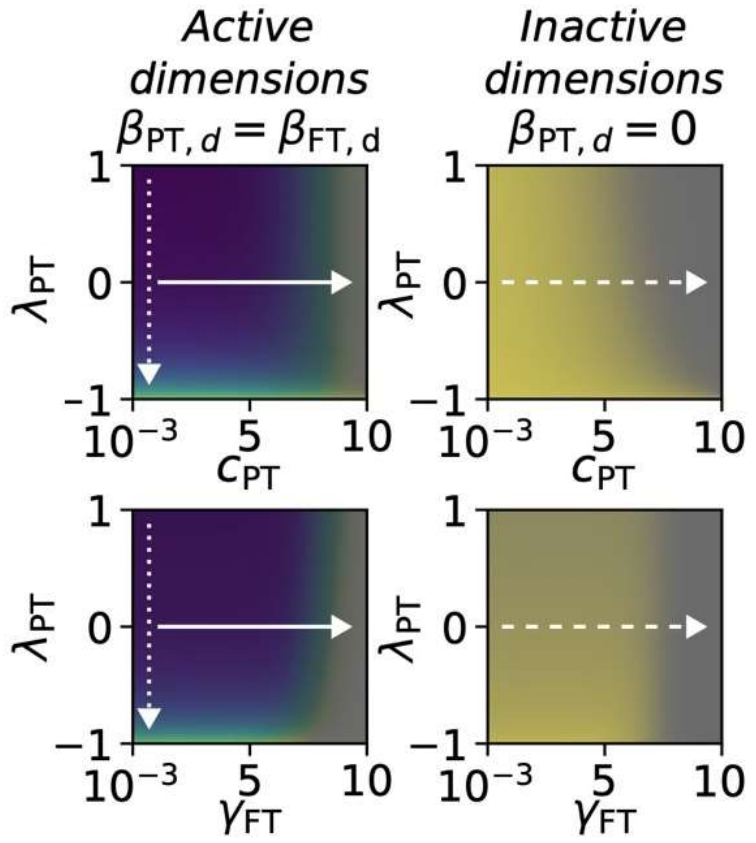


We fully characterise the **implicit bias** as a function of **relative scaling, scale and output rescaling** showcasing different learning regimes

1.2 The implicit bias of PT+FT

Fine-tuning





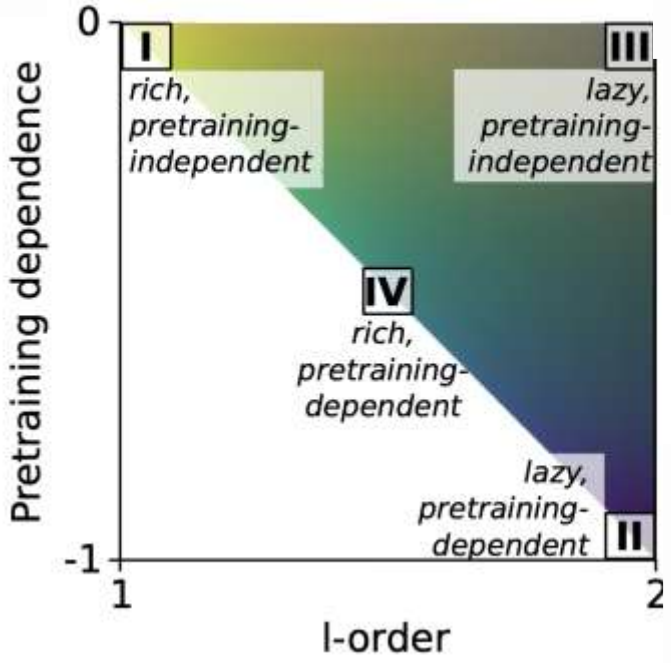
We fully characterise the **implicit bias** as a function of **relative scaling, scale and output rescaling** showcasing different learning regimes

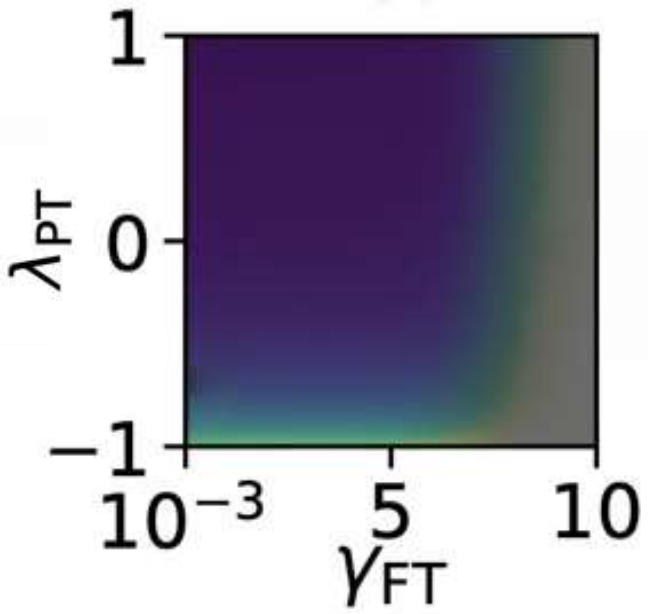
2. From implicit bias to generalization

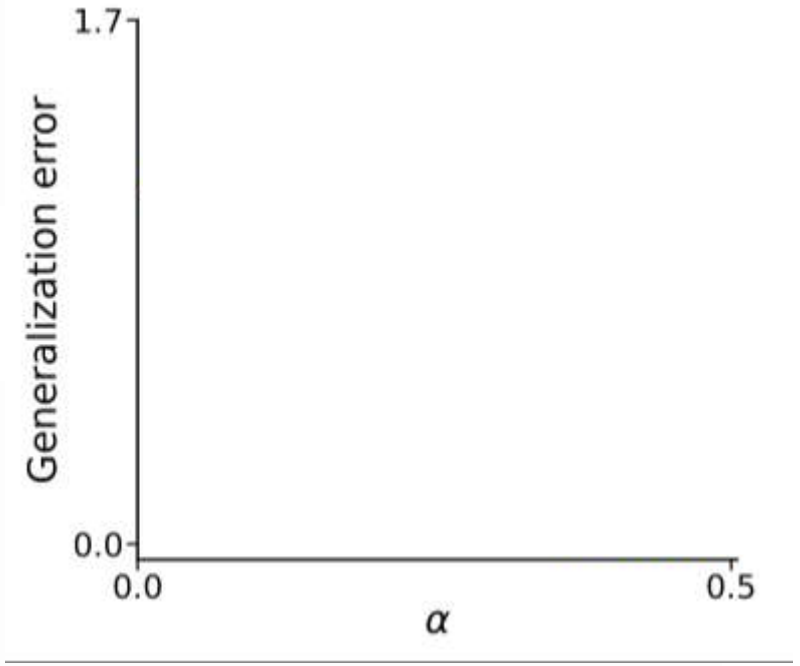
The right regime depends on sparsity and task overlap.

2. From implicit bias to generalization

Fine-tuning



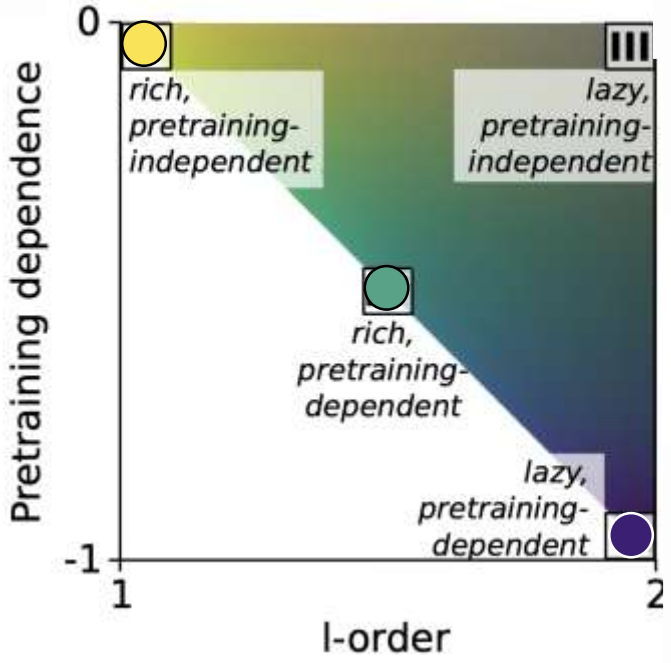


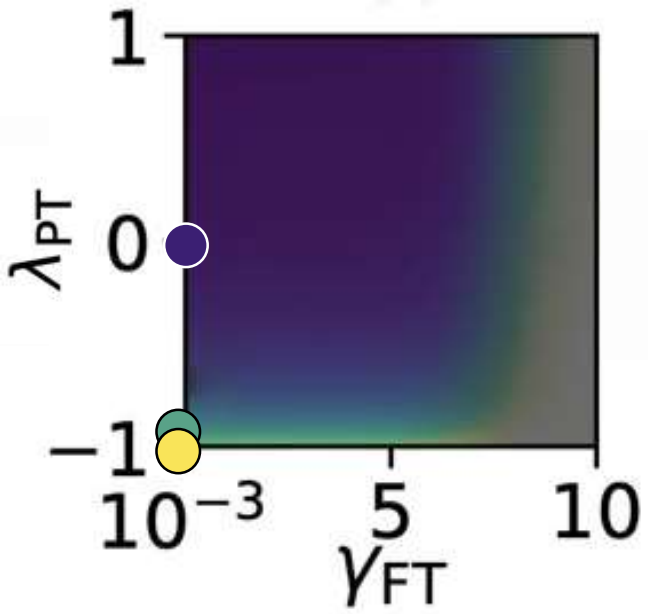


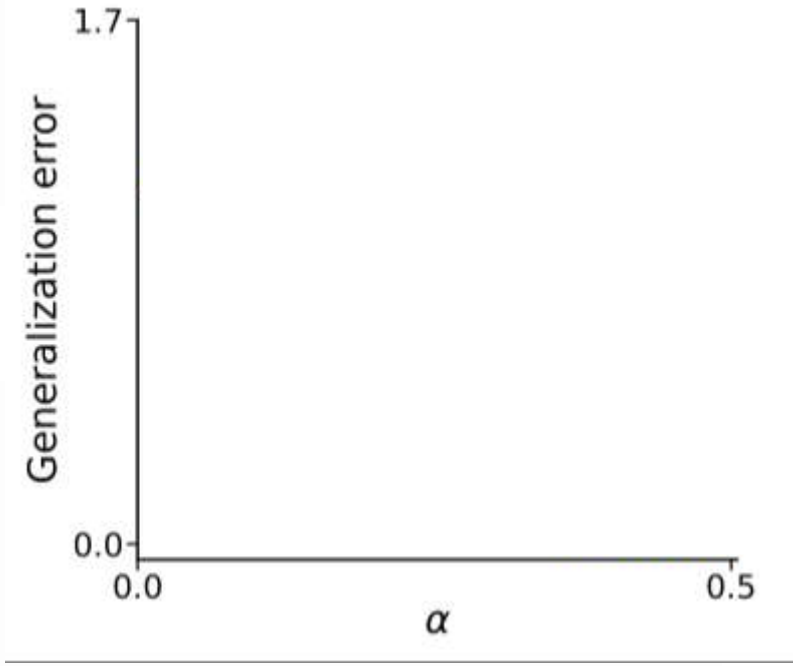
We fully characterise the generalisation error as a function of **initialisation parameters and task parameters** showcasing different learning regime.

2. From implicit bias to generalization

Fine-tuning



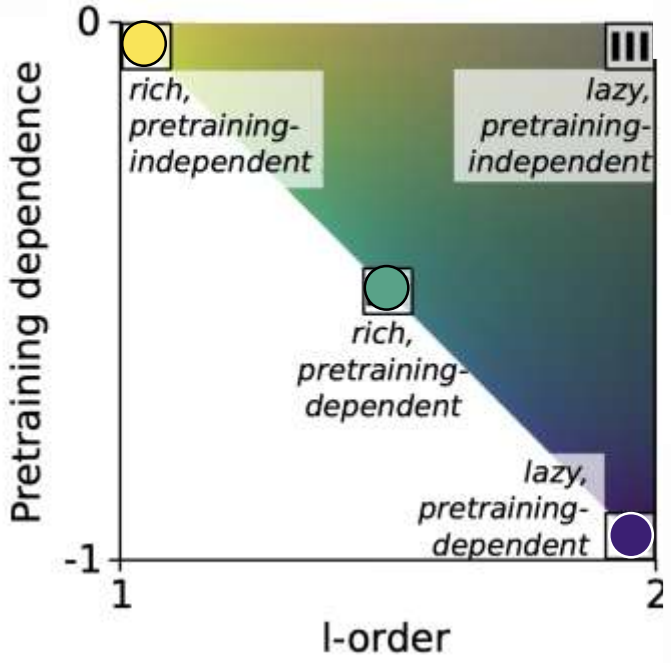


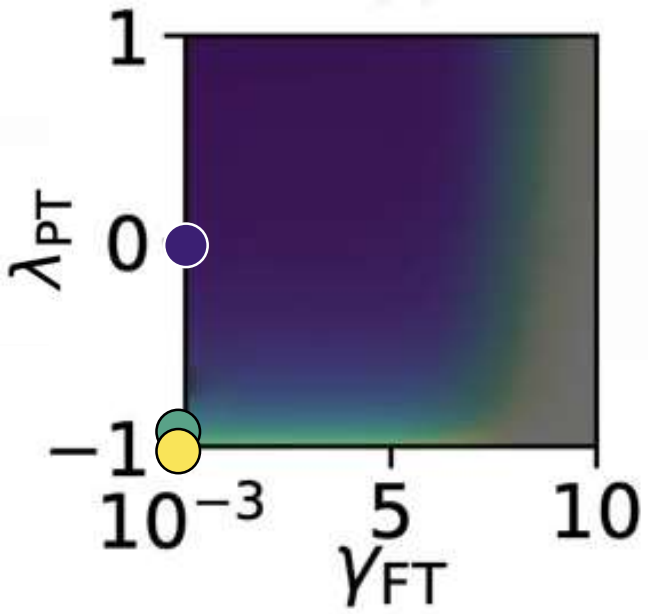


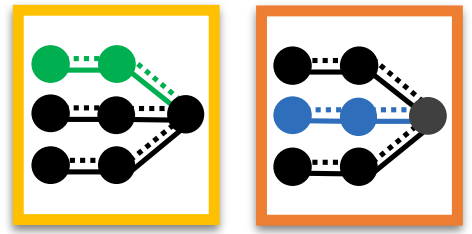
We fully characterise the generalisation error as a function of **initialisation parameters and task parameters** showcasing different learning regime.

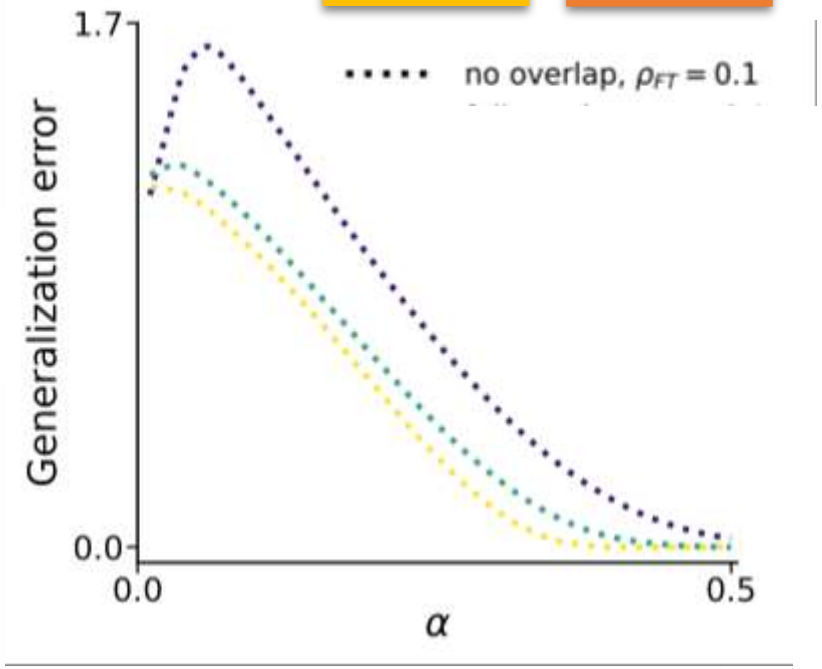
2. From implicit bias to generalization

Fine-tuning





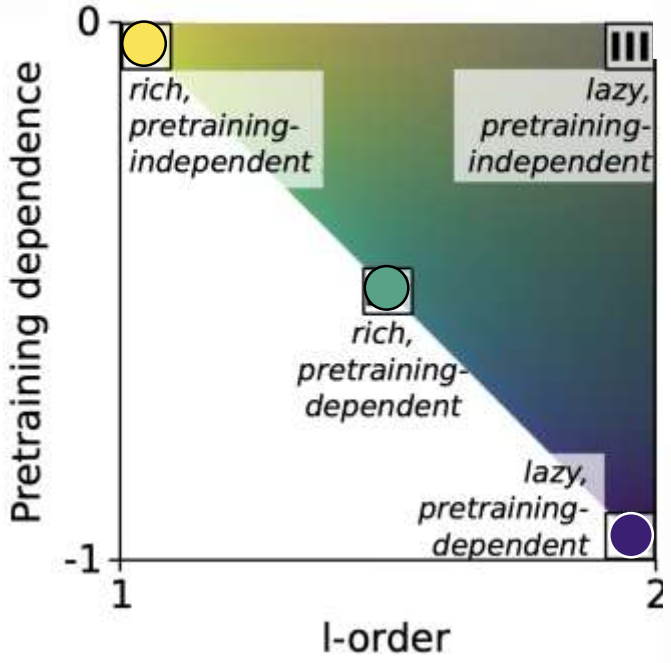


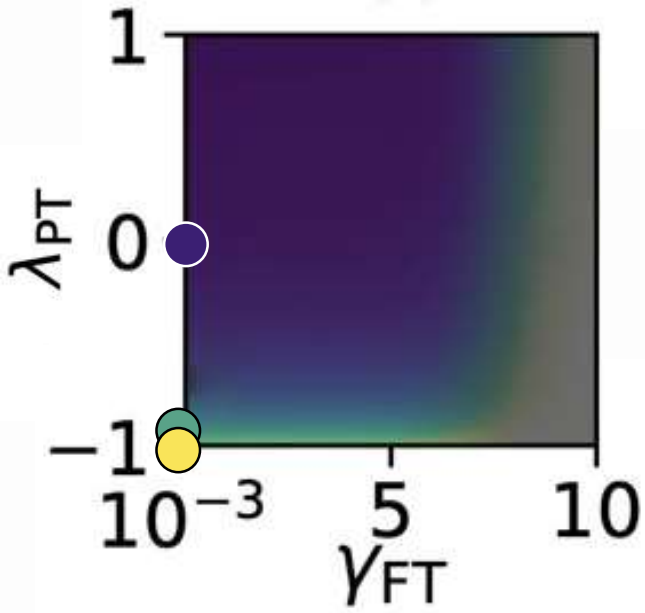


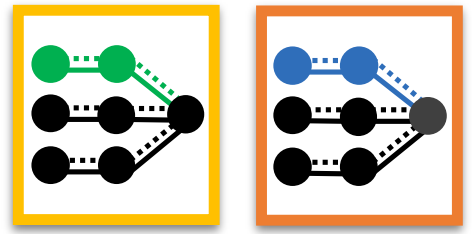
We fully characterise the generalisation error as a function of **initialisation parameters and task parameters** showcasing different learning regime.

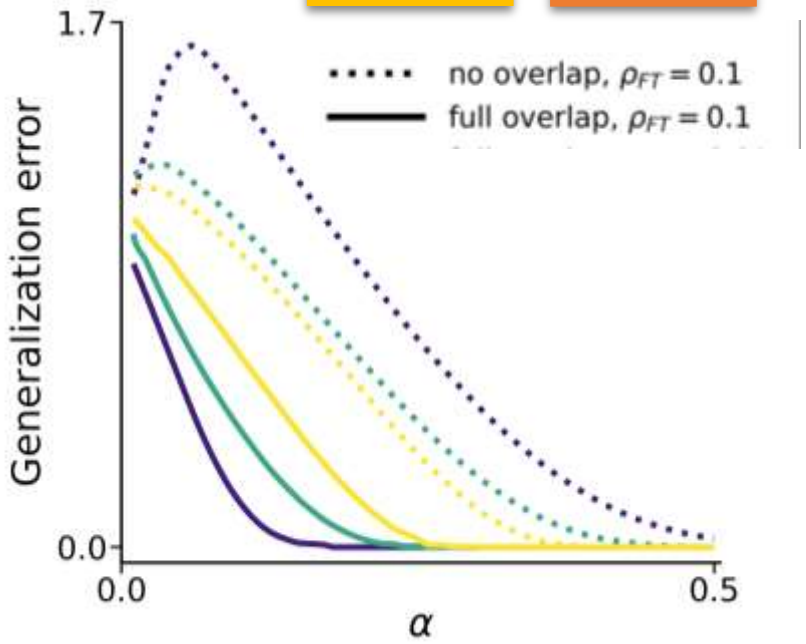
2. From implicit bias to generalization

Fine-tuning





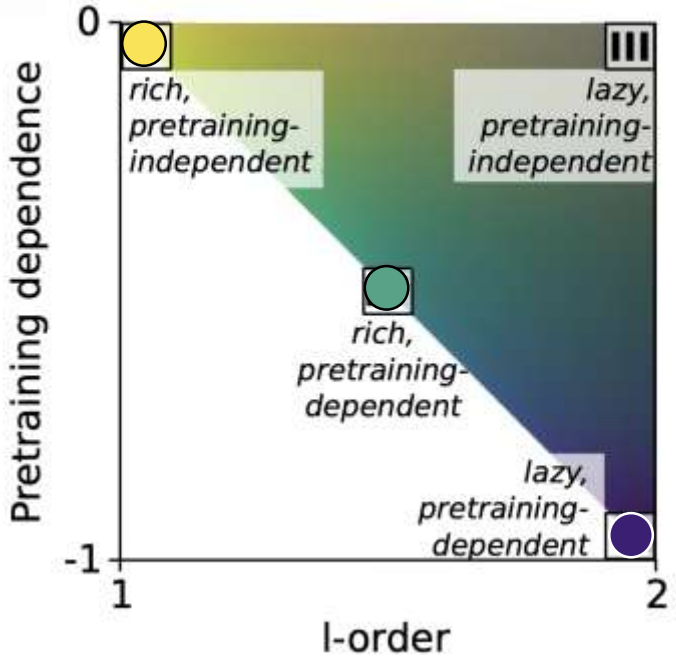


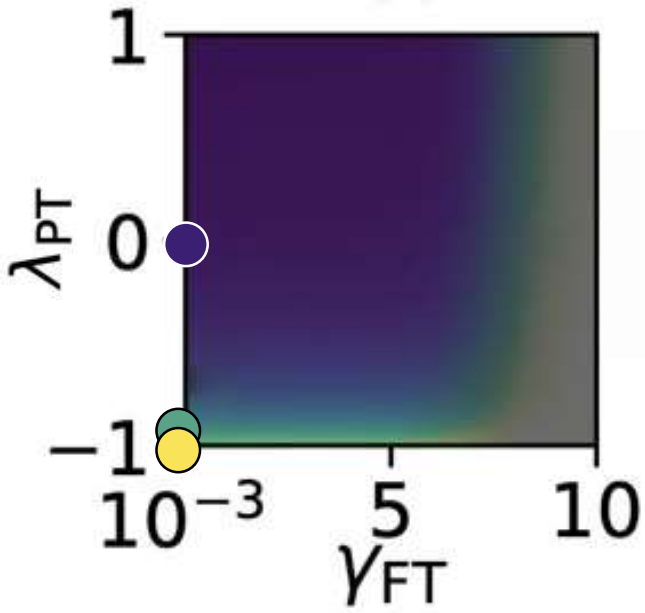


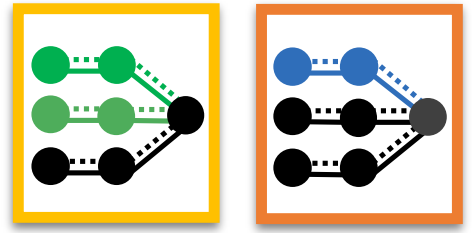
We fully characterise the generalisation error as a function of **initialisation parameters and task parameters** showcasing different learning regime.

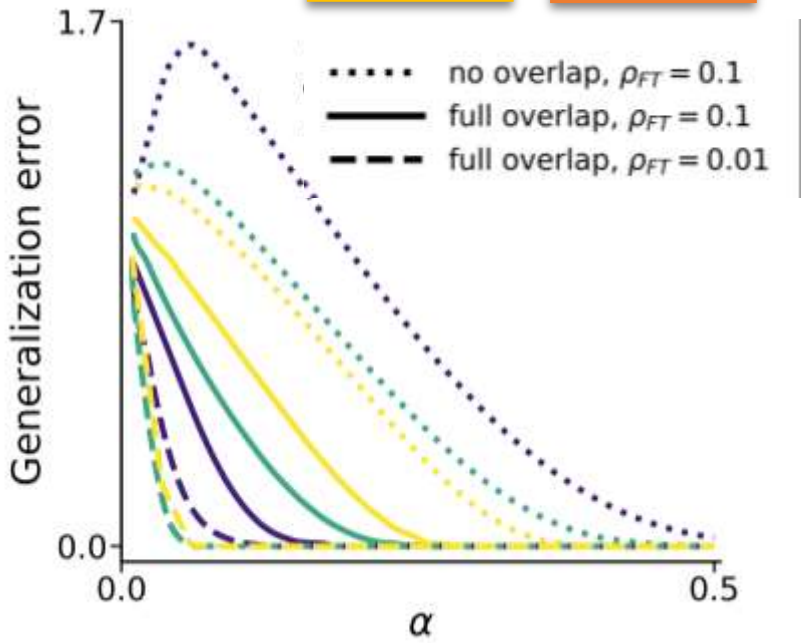
2. From implicit bias to generalization

Fine-tuning





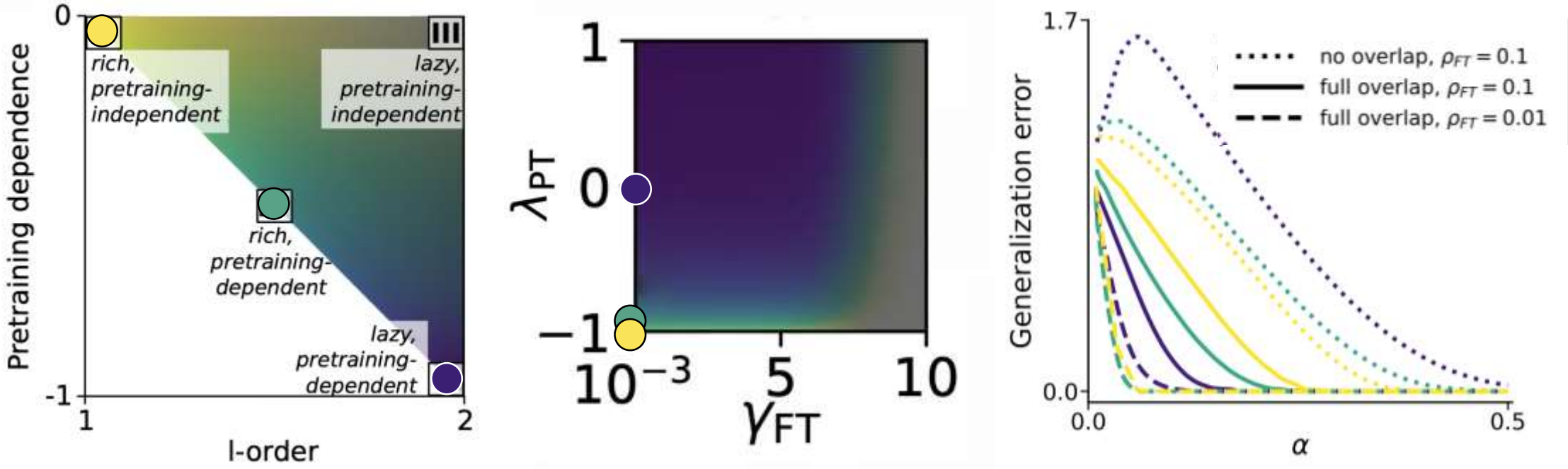




We fully characterise the generalisation error as a function of **initialisation parameters and task parameters** showcasing different learning regime.

2. From implicit bias to generalization

Fine-tuning



The **right regime** depends on **sparsity** and **task overlap.**

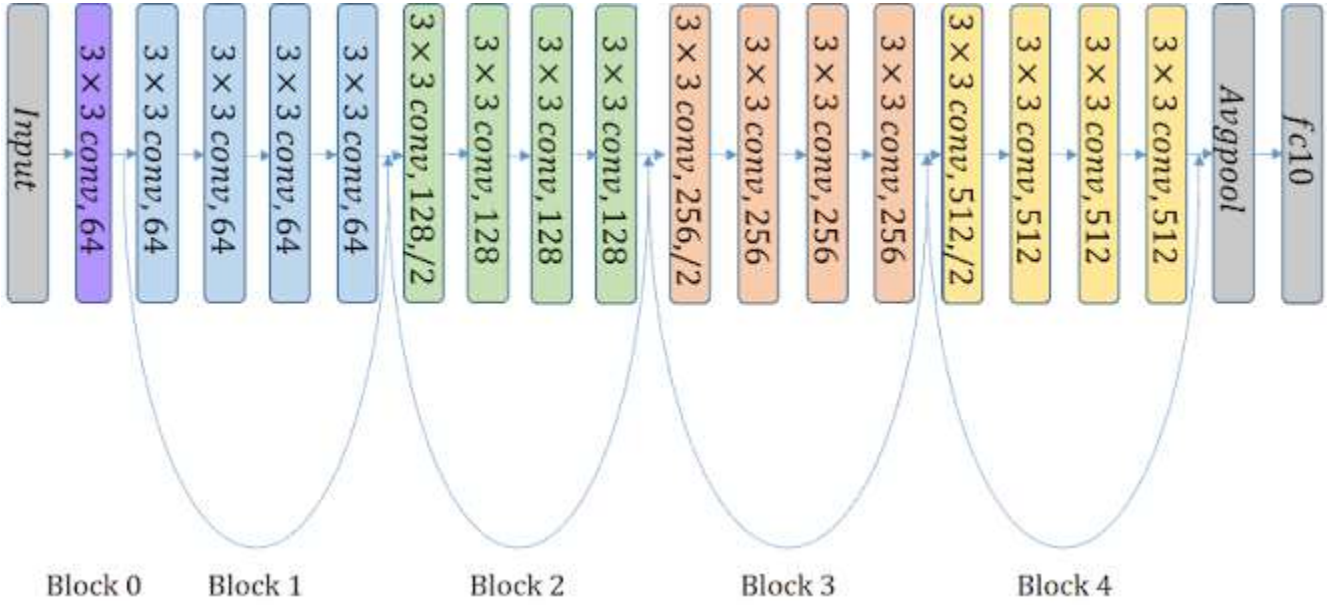
3. Do the same levers matter in practical networks?

A small ResNet/CIFAR test of the theoretical control knobs.

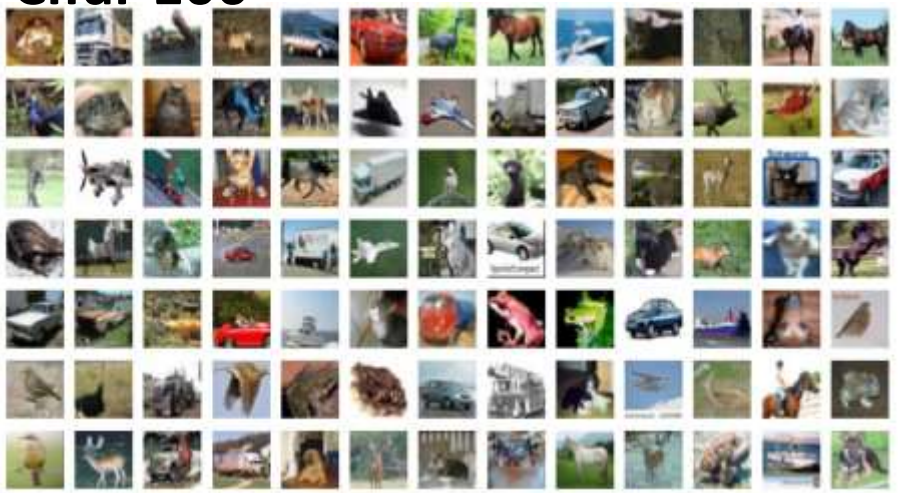
3. Do the same levers matter in practical networks?

ResNet18 Cifar100 experiment

ResNet 18

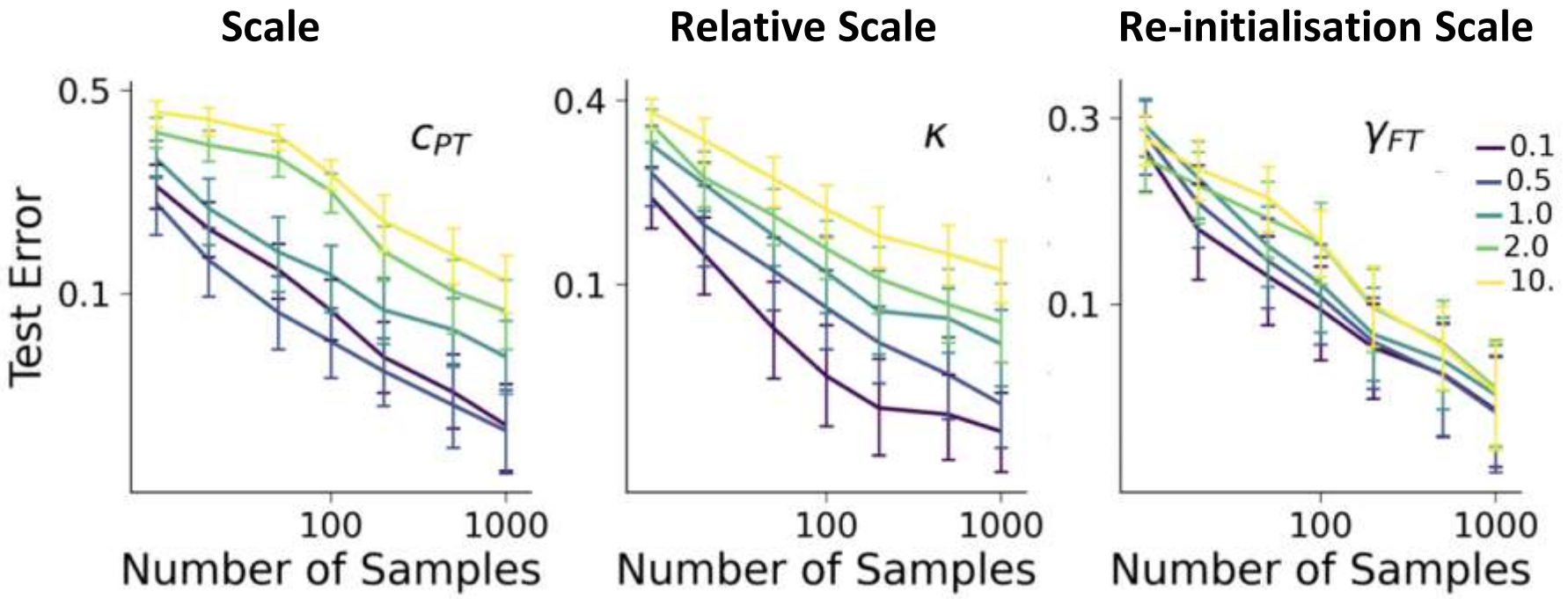


Cifar 100



3. Do the same levers matter in practical networks?

ResNet18 Cifar100 experiment



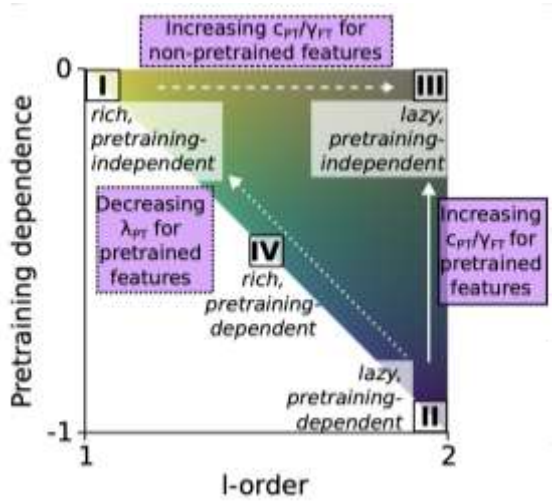
We show that the **relative scaling, scale and output rescaling** also impact the generalisation of fine-tuning in practice

4. Conclusion & Outlook

4. Conclusion and Outlook

We analytically derive **the implicit bias of PT+FT in diagonal linear networks** and the **resulting generalisation error** as a fu

- Pretraining weight initialisation
- Output weight initialisation
- Task structure (sparsity and overlap)

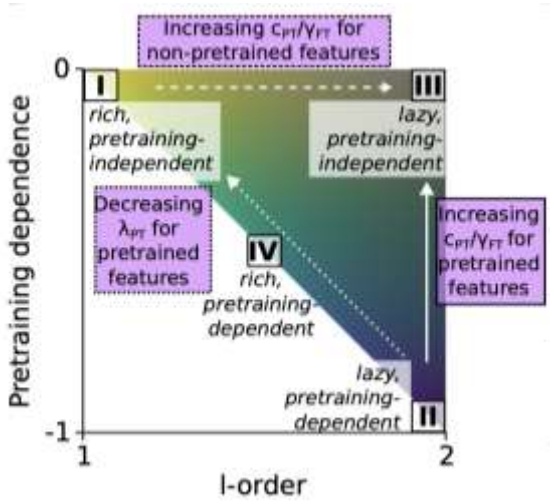


Fine-tuning induces **four distinct regimes** in diagonal linear networks.

4. Conclusion and Outlook

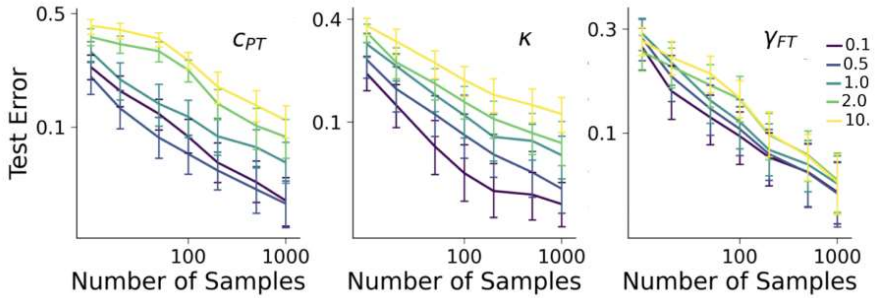
We analytically derive **the implicit bias of PT+FT in diagonal linear networks** and the **resulting generalisation error** as a fu

- Pretraining weight initialisation
- Output weight initialisation
- Task structure (sparsity and overlap)



Fine-tuning induces **four distinct regimes** in diagonal linear networks.

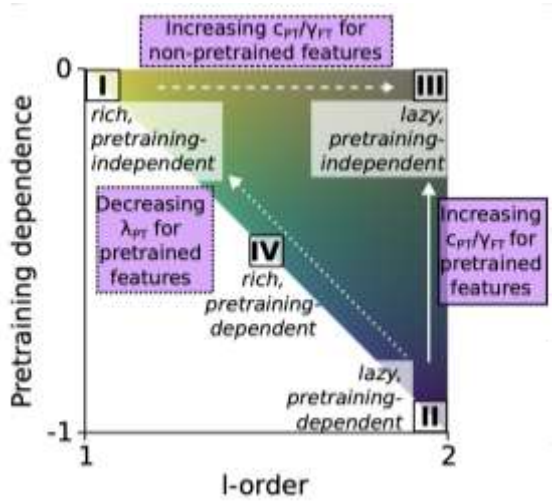
We show that the same parameters also impact the degree of feature learning in fine-tuning **non-linear network.**



4. Conclusion and Outlook

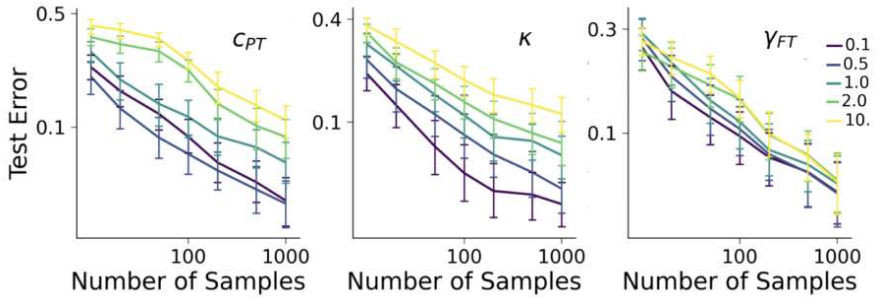
We analytically derive **the implicit bias of PT+FT in diagonal linear networks** and the **resulting generalisation error** as a fu

- Pretraining weight initialisation
- Output weight initialisation
- Task structure (sparsity and overlap)



Fine-tuning induces **four distinct regimes** in diagonal linear networks.

We show that the same parameters also impact the degree of feature learning in fine-tuning **non-linear network.**

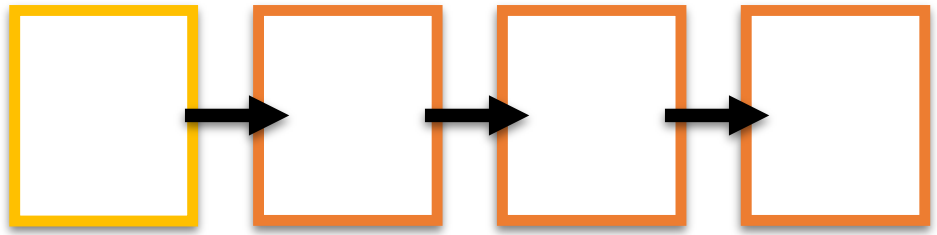


The best regime is not universal: it depends on pretraining characteristics and the task overlap and sparsity.

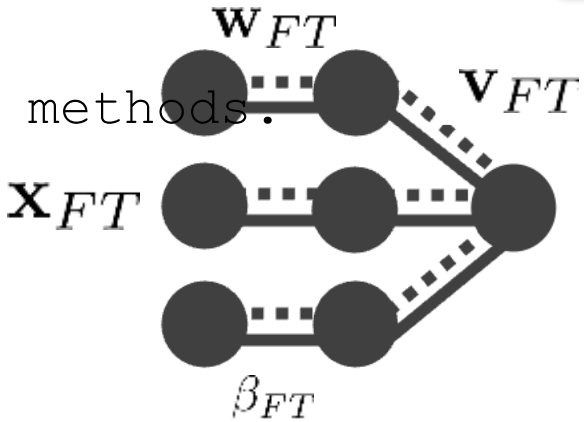
4. Conclusion and Outlook

Future directions

- Continual/Curriculum learning across multiple tasks.
- Unlearning.

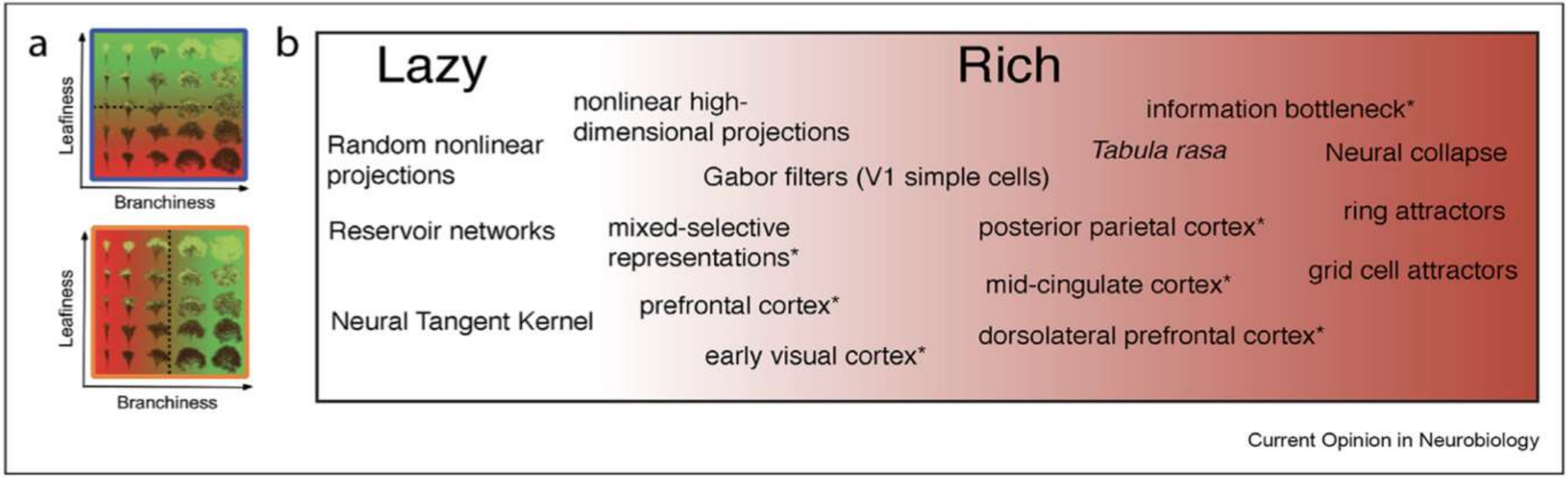


- Different fine-tuning methods.



$$\mathbf{v}^{\pm}_{\mathbf{FT}}(0) = \gamma_{FT}\mathbf{v}^{\pm}_{\mathbf{PT}}(0)$$

4. Conclusion and Outlook



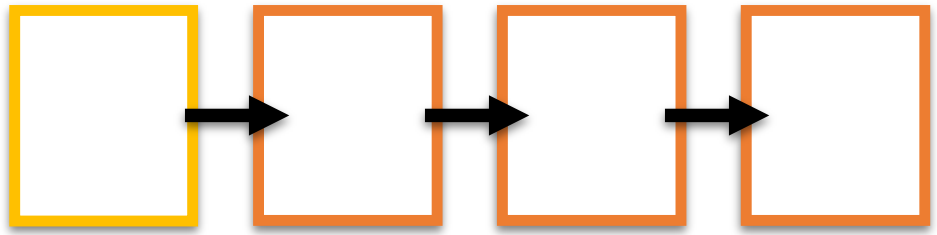
Broader view

- Understanding learning regime across more complex setups
  - In **neuroscience** .

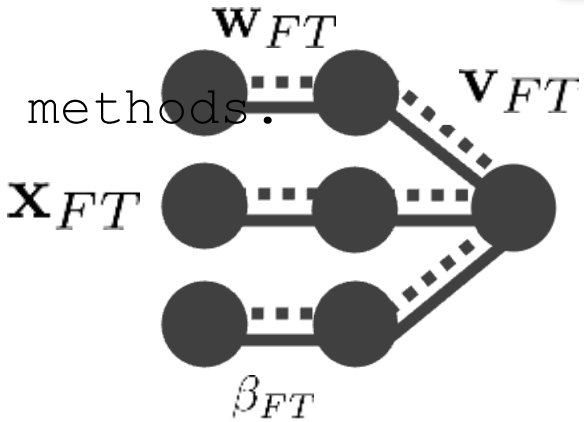
4. Conclusion and Outlook

Future directions

- Continual/Curriculum learning across multiple tasks.
- Unlearning.



- Different fine-tuning methods.



$$\mathbf{v}^{\pm}_{\mathbf{FT}}(0) = \gamma_{FT}\mathbf{v}^{\pm}_{\mathbf{PT}}(0)$$

Broader view

- Understanding learning regime across more complex setups
  - In **neuroscience**
  - In **machine learning**






Thank you!

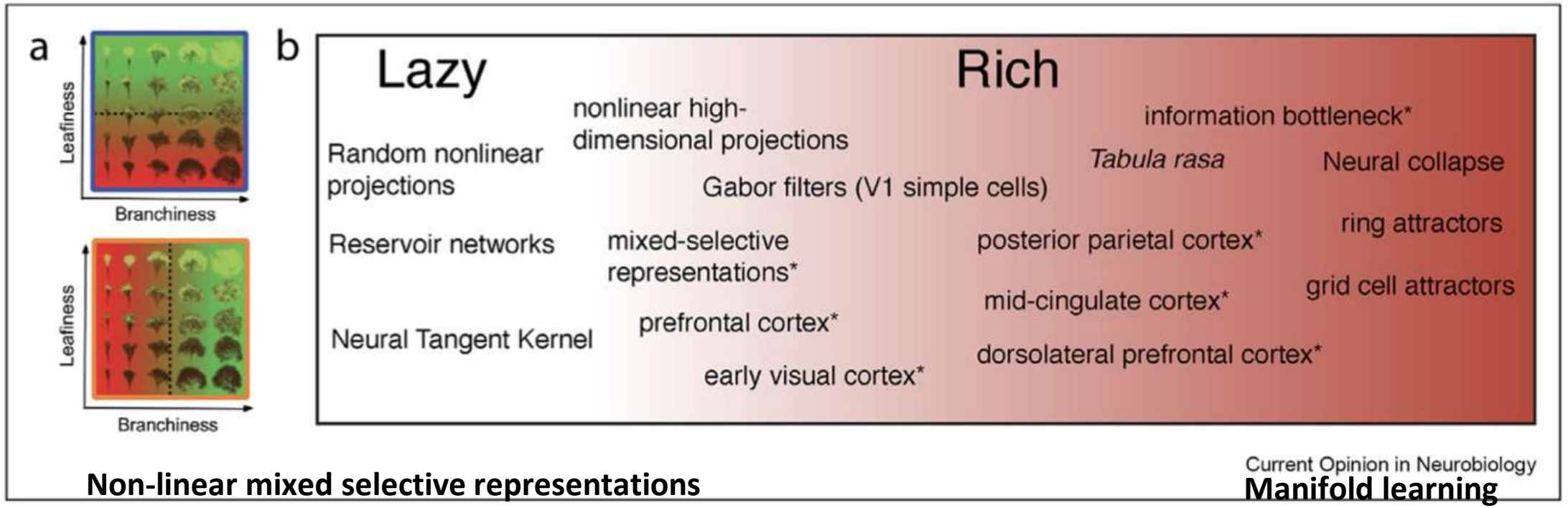
Any Questions?

Left to right, top to bottom: Nicolas Anguita Samuel Lippl, Francesco Locatello, Marco Mondelli, Andrew Saxe, Flavia Mancini

Outstanding questions

- Can we measure l-order and feature dependence in large-scale networks?
- Does an inverse correlation between the two measures hold in large-scale networks as well?
- Can we induce nested feature learning through explicit regularization?
- **Is the nested feature learning regime relevant for learning in humans and animals as well?**
- Can we use explicit weight regularization to theoretically understand the impact of initialization in other scenarios as well?

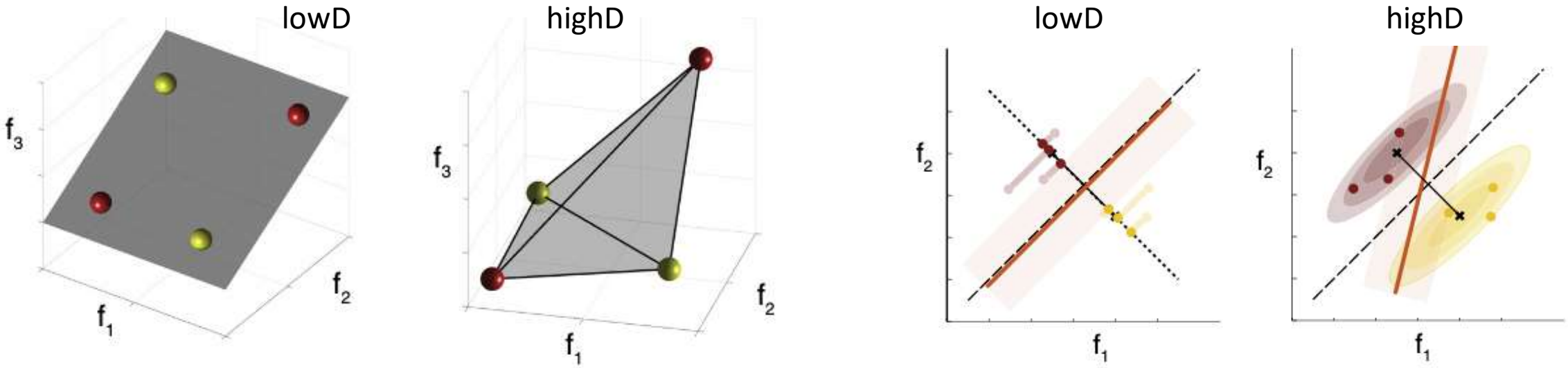
Background: Rich and Lazy learning regime in neuroscience



Lazy regime → **Rich regime**

[Rigotti et al. 2013, Raposo et al. 2014,., Tang et al. 2019, Flesch et al. 2022, Farrell et al. 2023, Boyle, et al 2024, Courellis et al. 2024, Nogueira et a.l 2022]

Background: Rich and Lazy learning regime in neuroscience



High dimensional neural codes imply greater separability

Low dimensional neural codes imply greater generalisability

[Rigotti et al. 2013, Raposo et al. 2014,., Tang et al. 2019, Flesch et al. 2022, Farrell et al. 2023, Boyle, et al 2024, Courellis et al. 2024, Nogueira et a.l 2022]

Application

Now, from Azulay et al. (2021), we know that the implicit bias of a diagonal linear network is parameterized as in Eq. (1) that converges to a zero loss solution can be expressed as:

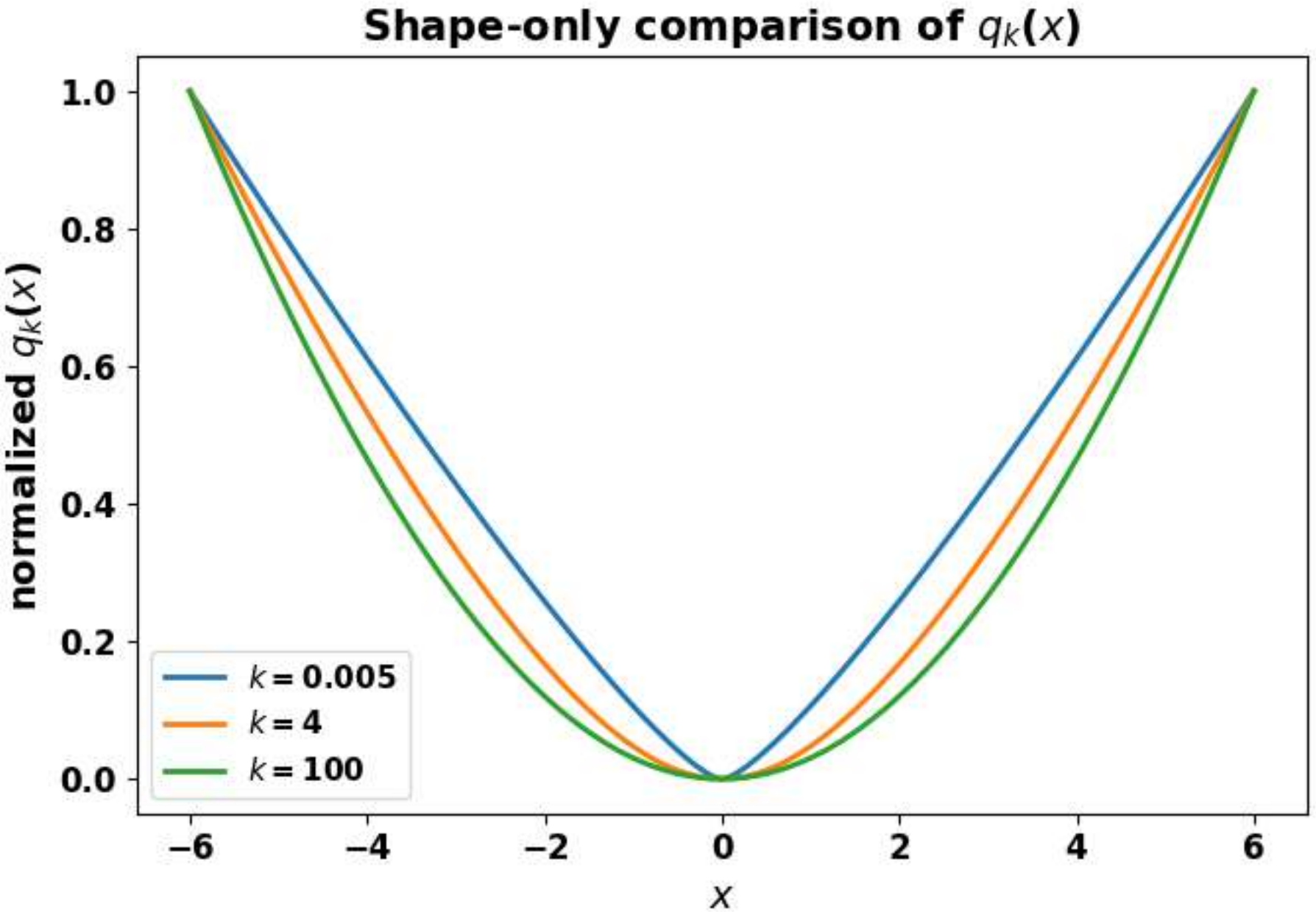
$$\beta_*(\infty) = \arg\min_{\beta_{FT}} Q_k(\beta_{FT}) \quad \text{s.t.} \quad \mathbf{X}_{FT}^\top \beta_{FT} = \mathbf{y}_{FT},$$

$$\text{where} \quad Q_k(\beta_{FT}) = \sum_{i=1}^{D} q_{k_i}(\beta_{FT,i}),$$

$$\text{with} \quad q_k(z) = \frac{\sqrt{k}}{4}\left(1 - \sqrt{1 + \frac{4z^2}{k}} + \frac{2z}{\sqrt{k}} \operatorname{arcsinh}\left(\frac{2z}{\sqrt{k}}\right)\right),$$

where $k_i = 4c_i^2$. In our case $k_i = 4c_{FT,i}^2 = \left(2(\tilde{\lambda}_{\text{PT},i} + c_{\text{PT},i})\left(1 + \sqrt{1 + (\beta_{\text{PT},i}/c_{\text{PT},i})^2}\right) + \gamma_{FT}^2\right)^2$. Hence $k_i = 4c_{FT,i}^2$. The theorem follows from replacing $\tilde{\lambda}_{\text{PT},i} = \lambda_{PT,i} c_{PT,i}$. □

Application



Application

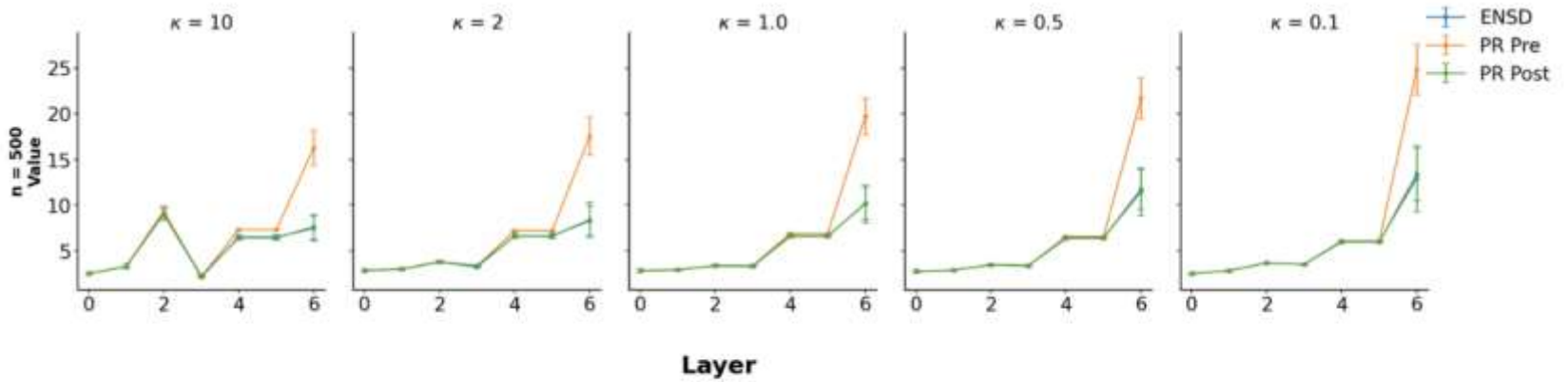


Figure 6. **ResNet experiments on CIFAR-100.** Resnet layers before and after fine-tuning (PR Pre and PR Post) as well as their ENSD as a function of the κ_{FT} re-initialization.

Application

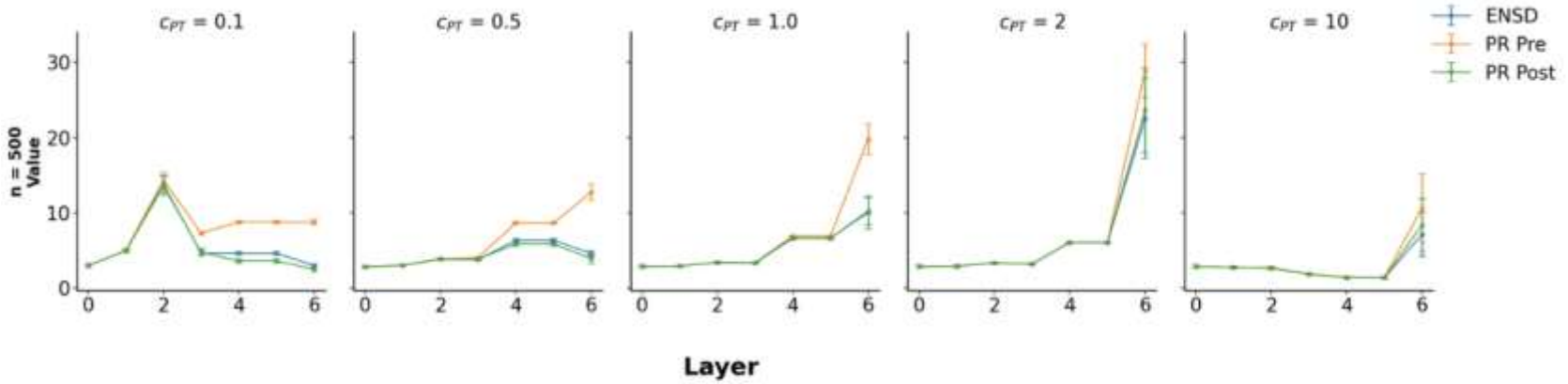


Figure 7. **ResNet experiments on CIFAR-100.** Resnet layers before and after fine-tuning (PR Pre and PR Post) as well as their ENSD as a function of the c_{PT} re-initialization.

Application

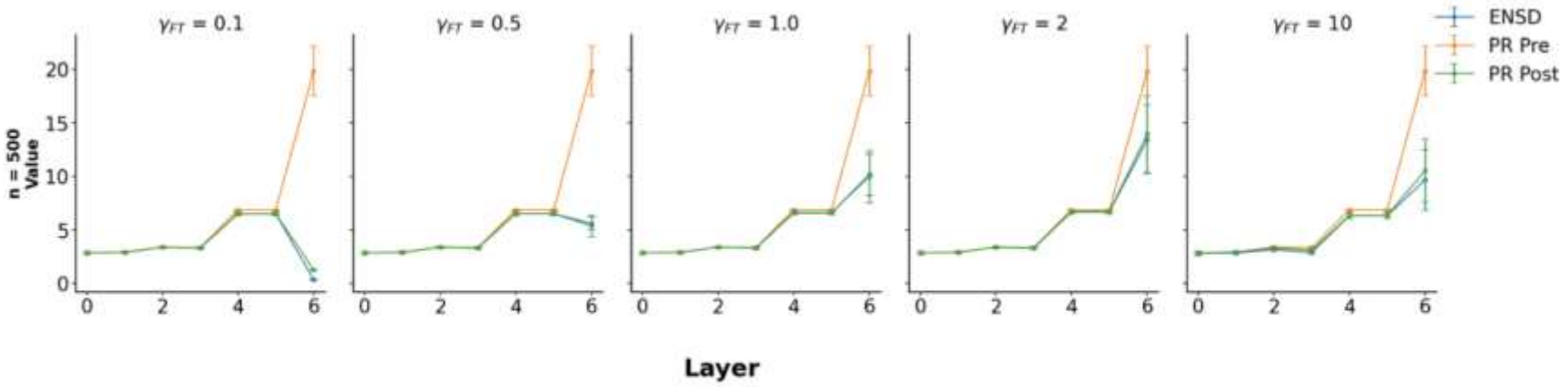


Figure 8. **ResNet experiments on CIFAR-100.** Resnet layers before and after fine-tuning (PR Pre and PR Post) as well as their ENSD as a function of the γ_{FT} re-initialization.

Application

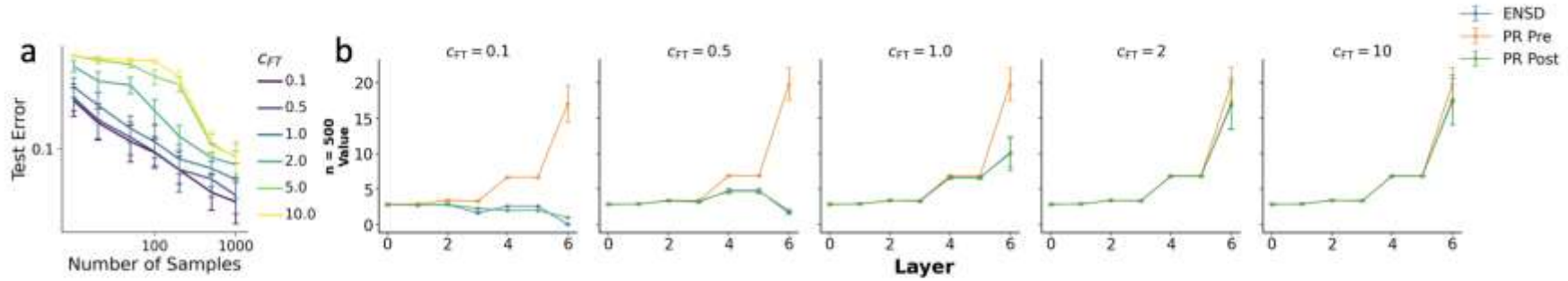
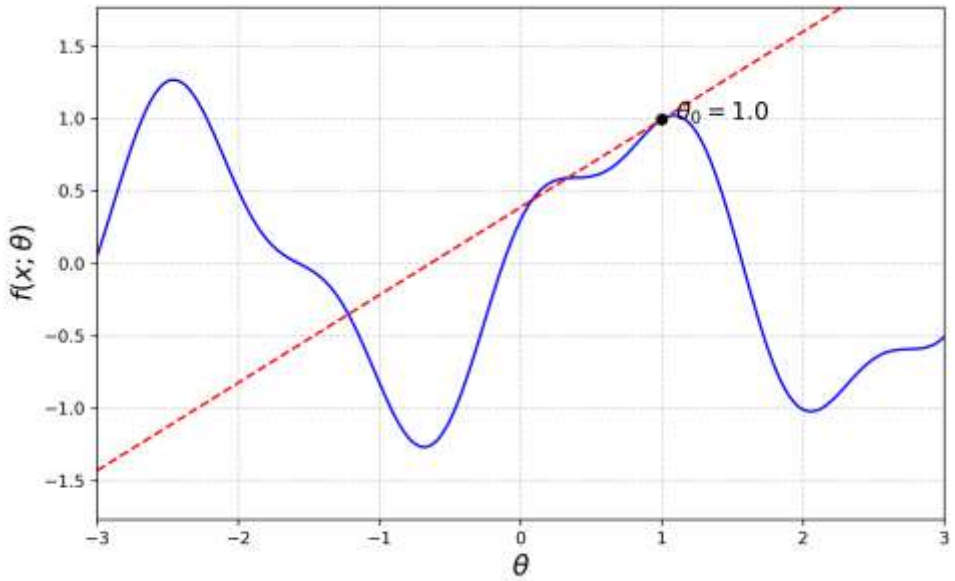


Figure 9. **ResNet experiments on CIFAR-100.** (a) Generalization performance as a function of the number of samples and initalization parameters. We vary c_{FT}. (b) Resnet layers before and after fine-tuning (PR Pre and PR Post) as well as their ENSD as a function of the γ_{FT} re-initialization.

Generalization in neural networks (Taylor's Version)

Approach Approximate neural networks by their first-order Taylor approximation

("**neural tangent kernel**", NTK)



Initial weights: θ_0

Initial function: $f(x;\theta_0)$

Initial derivative: $\partial_\theta f(x;\theta_0)$

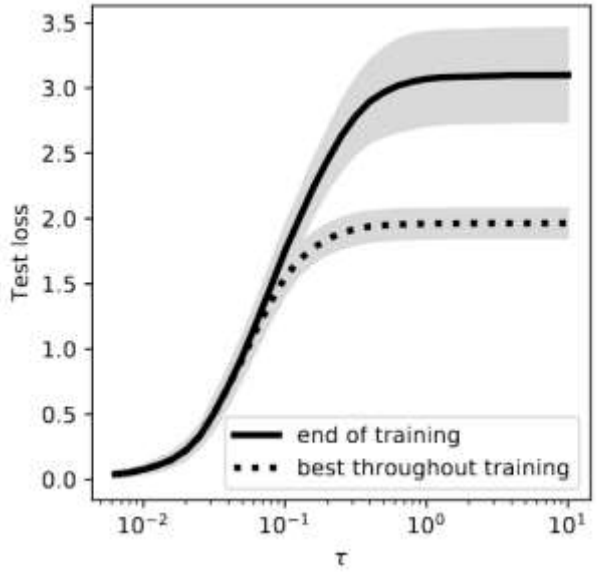
Taylor approximation: $\tilde{f}(x;\theta) = f(x;\theta_0) + \partial_\theta f(x;\theta_0)(\theta - \theta_0)$

Large initial weights

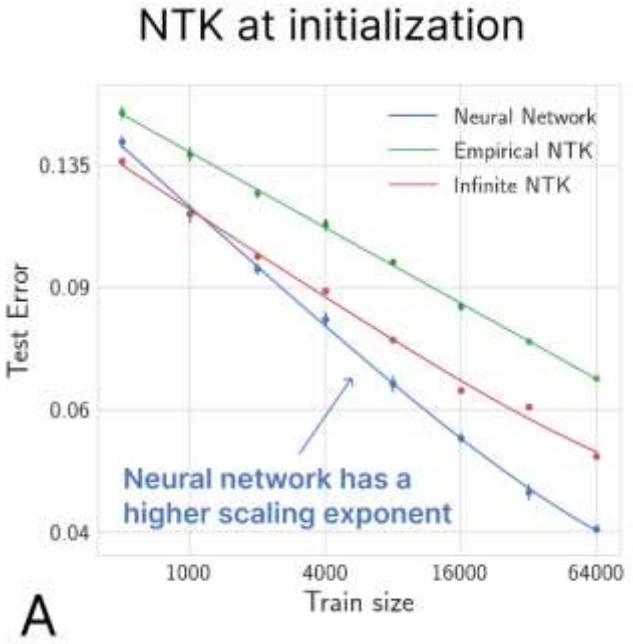
⇒ neural networks are approximated by the NTK throughout training

NTK cannot explain generalization in neural networks

Simple ReLU networks generalize better from small weights (Chizat et al., 2019)



Neural networks perform better on real-world tasks than their NTK approximation (Vyas et al., 2023)

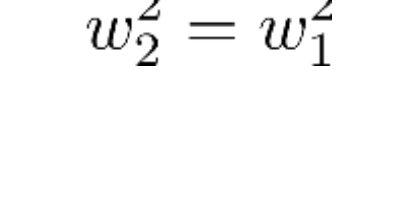


A Minimal Model of Pretraining and Fine-tuning

Set-up

Scalar

$$w_1^2 - w_2^2 = \lambda$$



$\lambda \to -\infty$ | $\lambda = 0$ | $\lambda \to +\infty$

	$\lambda \to -\infty$	$\lambda = 0$	$\lambda \to +\infty$
Scalar	$w_2^2 - w_1^2 = \lambda$	$w_2^2 = w_1^2$	$w_2^2 - w_1^2 = \lambda$
	$w_2^2 - w_1^2 = -5$		$w_2^2 - w_1^2 = 5$
	$w_2^2 <<< w_1^2$		$w_2^2 >>> w_1^2$

Lambda-balanced = relative scale

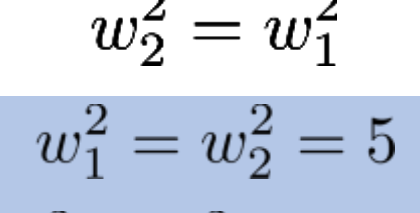
[Kunin et al. 2024, Dominé and Anguita et al. 2024]

A Minimal Model of Pretraining and Fine-tuning

Set-up

Scalar

$$w_1^2 - w_2^2 = \lambda$$



$\lambda \to -\infty$ | $\lambda = 0$ | $\lambda \to +\infty$

Scalar

$$w_2^2 - w_1^2 = \lambda \qquad w_2^2 = w_1^2 \qquad w_2^2 - w_1^2 = \lambda$$

$$w_1^2 = w_2^2 = 5$$

$$w_1^2 = w_2^2 = 0.01$$

relative scale $\neq$ **scale**

[Kunin et al. 2024, Dominé and Anguita et al. 2024]