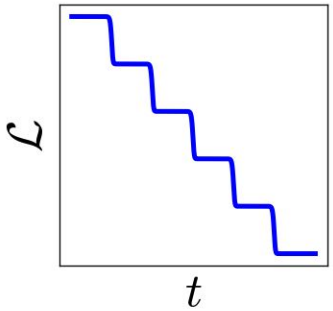
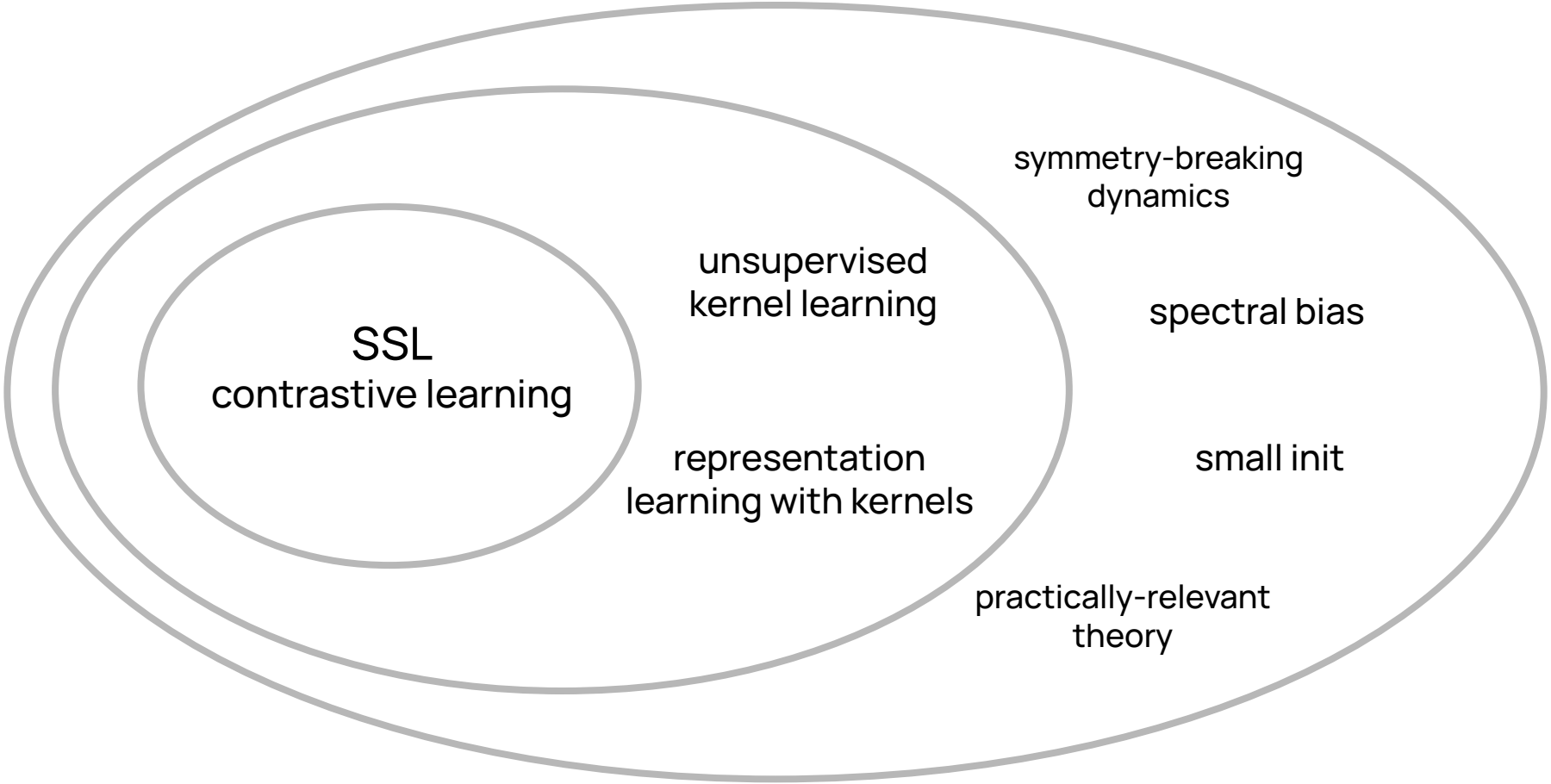
On the stepwise nature of SSL



Jamie Simon

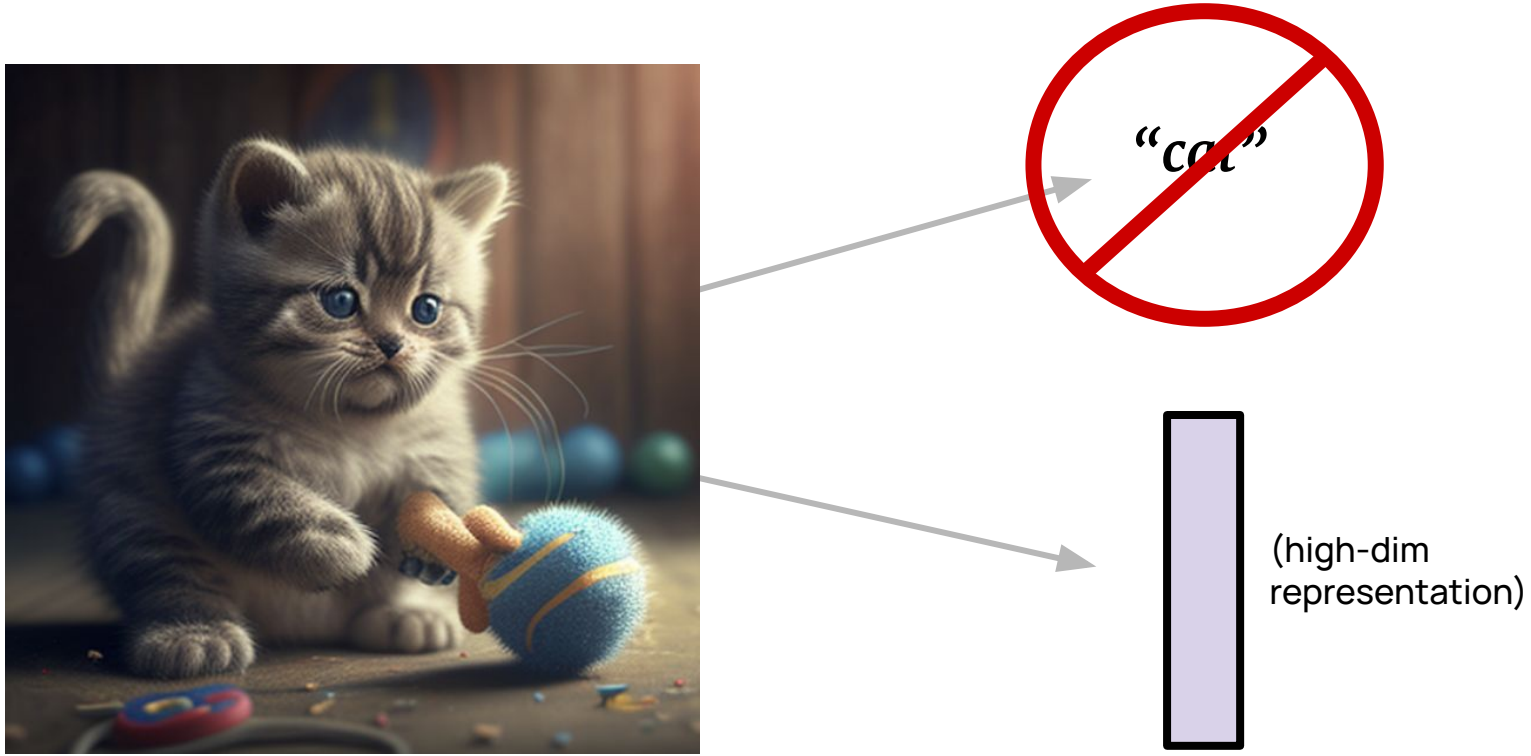
with Maksis Knutins, Liu Ziyin, Daniel Geisz, Abe Fetterman, and Josh Albrecht

paper: tinyurl.com/stepwise-ssl-paper

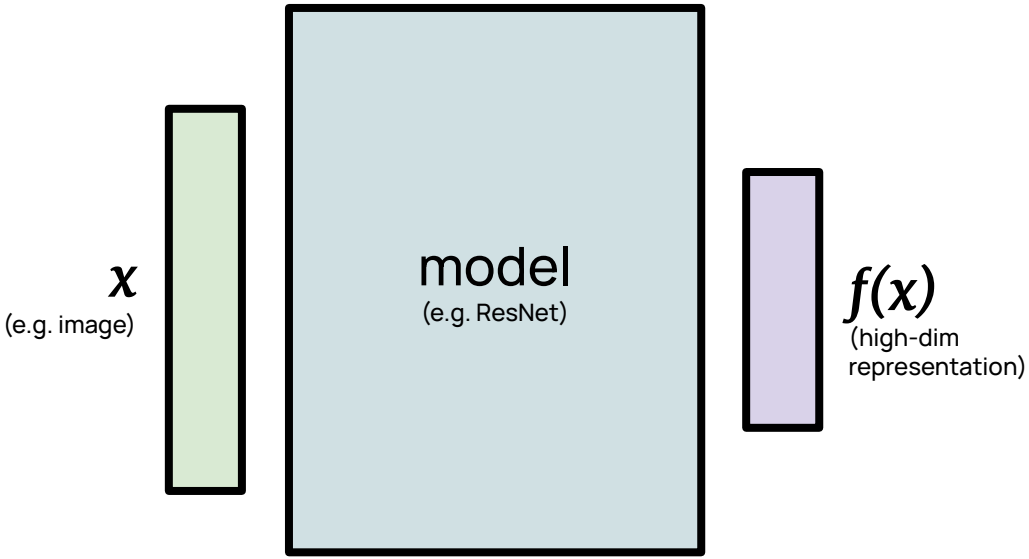

symmetry-breaking
dynamics
unsupervised
kernel learning
spectral bias
SSL
contrastive learning
representation
learning with kernels
small init
practically-relevant
theory

As theorists, we ought to look beyond supervised learning.

We should think about learning *representations*, not *target functions*.



Joint embedding SSL learns representations with a simple prescription.



let x, x' be two related samples.

goal of training: **make $f(x)$ close to $f(x')$**

(but not degenerate)

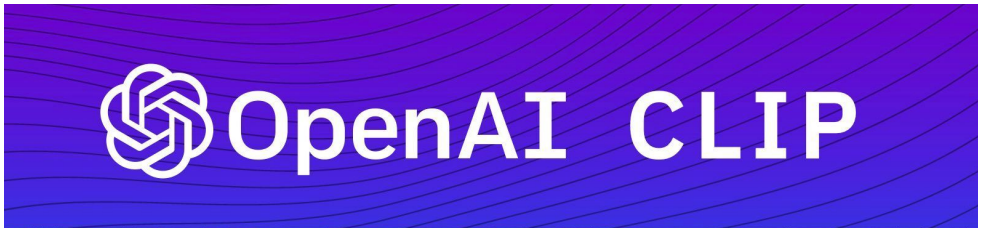
Surprisingly, this works very well…

SimCLR, Barlow Twins, VICReg, BYOL, DINO, CLIP

A Simple Framework for Contrastive Learning of Visual Representations

---

Ting Chen [1] Simon Kornblith [1] Mohammad Norouzi [1] Geoffrey Hinton [1]



SSL is state-of-the-art in image domains.

…but here’s what we understand about it:

[this slide intentionally left blank]

slide reproduced without permission from Stephan (2023)

What is the nature of the learning process of SSL?

What representations are ultimately learned?

Idea: answer these Qs in the linear case and check agreement with deep nets.

Plan of attack

1. Find and solve an exactly solvable case of SSL with a **linear model.**
2. Kernelize it.
3. Check agreement with realistic SSL.

Linear model of **Barlow Twins** (Zbontar et al. 2021)

Let the dataset contain n "positive pairs" $(\mathbf{x}_i, \mathbf{x}'_i)_{i=1}^n$.

Let the trained function be linear as $\mathbf{f}(\mathbf{x}) = \mathbf{W}\mathbf{x}$.

Correlation matrix: $\boldsymbol{C} \equiv \frac{1}{2n} \sum_i \left(\boldsymbol{f}(\boldsymbol{x}_i) \boldsymbol{f}(\boldsymbol{x}'_i)^\top + \boldsymbol{f}(\boldsymbol{x}'_i) \boldsymbol{f}(\boldsymbol{x}_i)^\top \right)$

Barlow Twins loss: $\mathcal{L} = \|\boldsymbol{C} - \boldsymbol{I}_d\|_F^2$

Train with GD: $\frac{d\boldsymbol{W}}{dt} = -\nabla_{\boldsymbol{W}} \mathcal{L}.$

Key insight: diagonalize w.r.t. the “feature cross-covariance.”

$$\boldsymbol{\Gamma} \equiv \frac{1}{2n} \sum_i \left(\boldsymbol{x}_i \boldsymbol{x}_i'^\top + \boldsymbol{x}_i' \boldsymbol{x}_i^\top \right)$$

$$\Longrightarrow \qquad \mathcal{L} = \left\| \boldsymbol{W} \boldsymbol{\Gamma} \boldsymbol{W}^\top - \boldsymbol{I}_d \right\|_F^2$$

$$\Longrightarrow \qquad \frac{d\boldsymbol{W}}{dt} = -4 \left(\boldsymbol{W} \boldsymbol{\Gamma} \boldsymbol{W}^\top - \boldsymbol{I}_d \right) \boldsymbol{W} \boldsymbol{\Gamma}$$

^This is *exactly solvable* when $\boldsymbol{W_o}$ = $\boldsymbol{W(t=o)}$ is either *aligned* or *small*.

Aligned init: $\boldsymbol{W}_0 = \boldsymbol{U} \boldsymbol{S}_0 \boldsymbol{\Gamma}^{(\leq d)}$

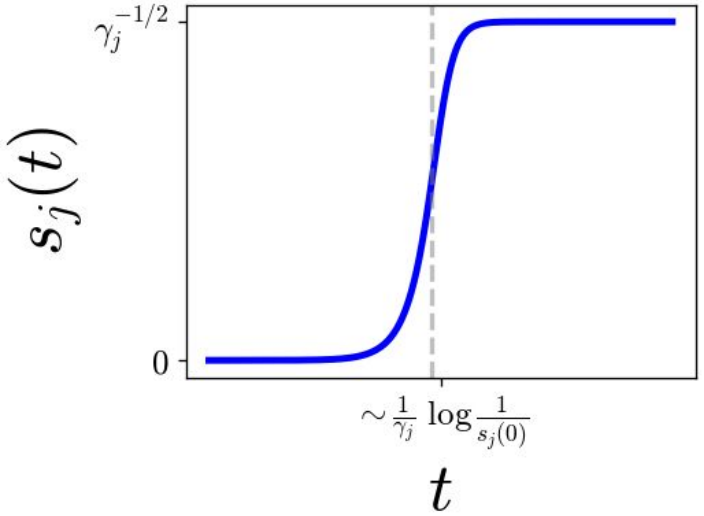
right singular vectors are
the top d eigenvectors of $\boldsymbol{\Gamma}$

From aligned init, each singular value evolves independently.

Proposition 4.1 (Trajectory of $\boldsymbol{W}(t)$ from aligned initialization). *If $\boldsymbol{W}(0) = \boldsymbol{W}_0$ as given by Equations 7 and 8, then*

$$\boldsymbol{W}(t) = \boldsymbol{U}\boldsymbol{S}(t)\boldsymbol{\Gamma}^{(\leq d)} \tag{9}$$

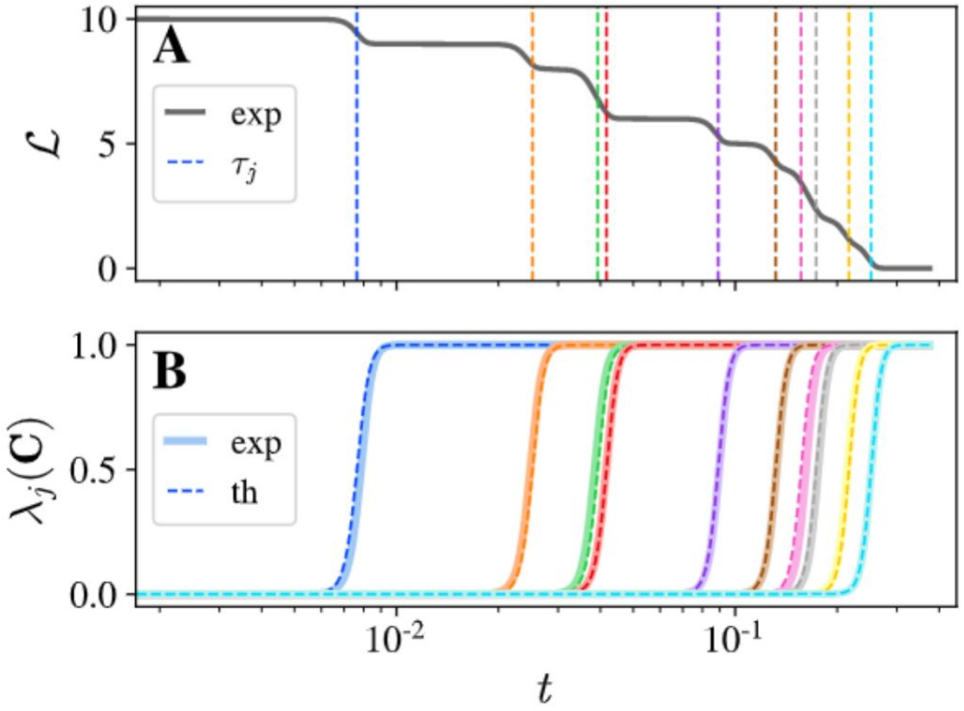
with $\boldsymbol{S}(t) = diag(s_1(t), ..., s_d(t))$ and



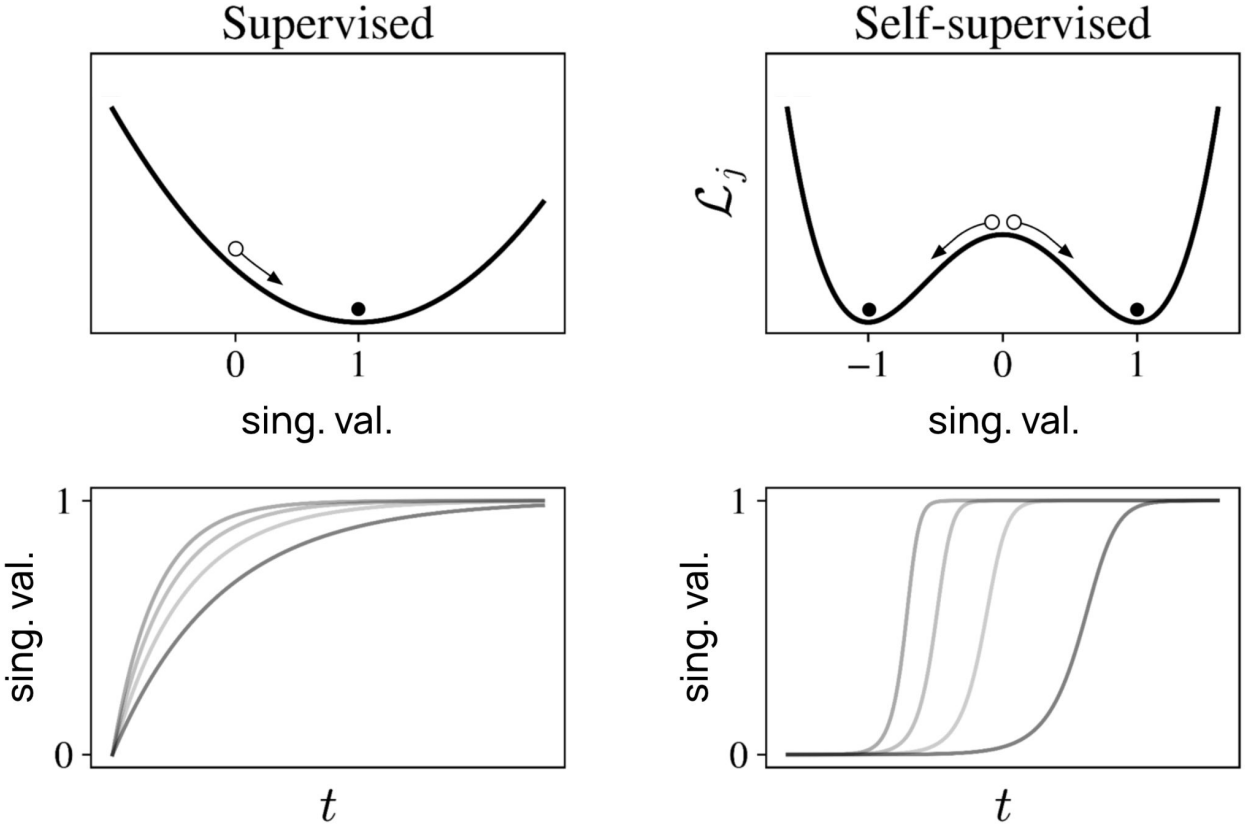
$$s_j(t) = \frac{e^{4\gamma_j t}}{\sqrt{s_j^{-2}(0) + (e^{8\gamma_j t} - 1)\gamma_j}}. \tag{10}$$

j-th eigenvalue of $\boldsymbol{\Gamma}$

From *generic small init*, these dynamics follow the aligned trajectory!



Representation learning is a process of *symmetry breaking.*



$$\dot{s_j} = \gamma_j(s_j^* - s_j)$$

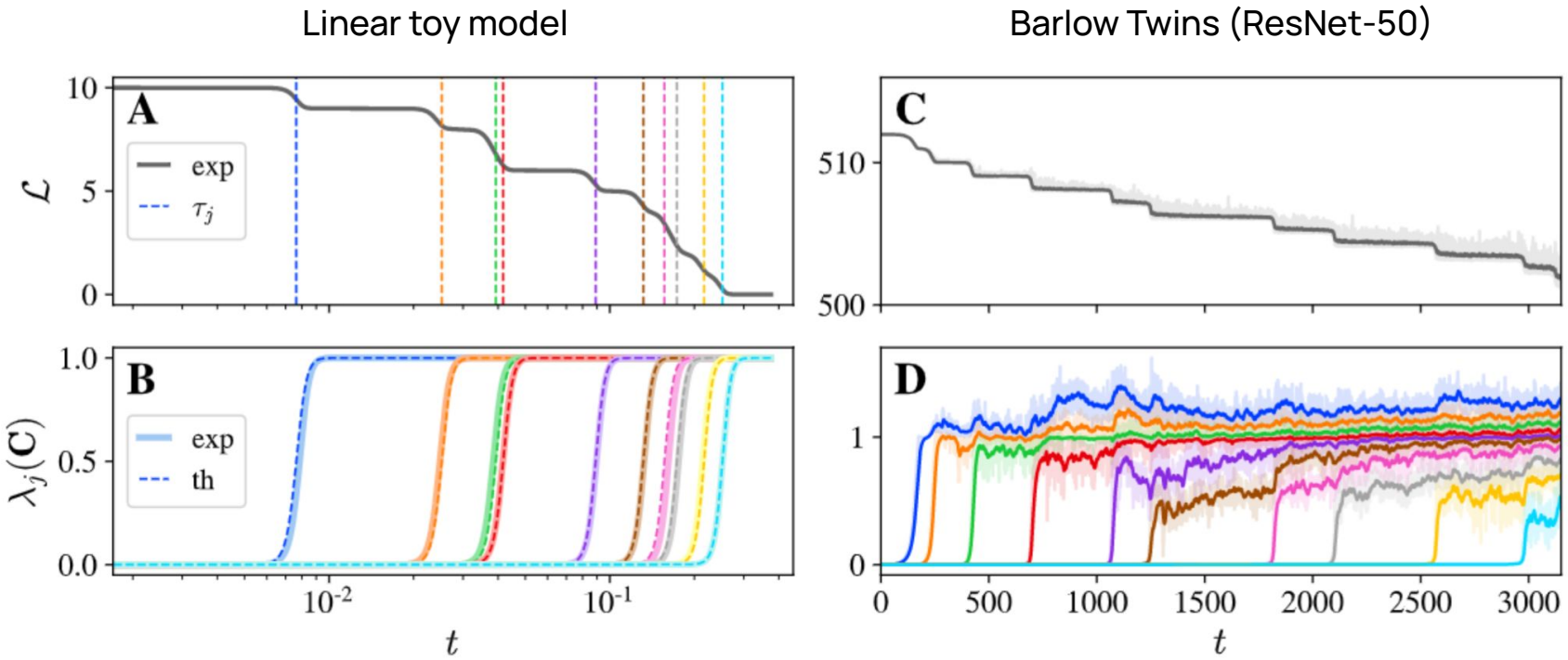
$$\dot{s_j} = \gamma_j s_j(1 - s_j^2)$$

It's kernel time

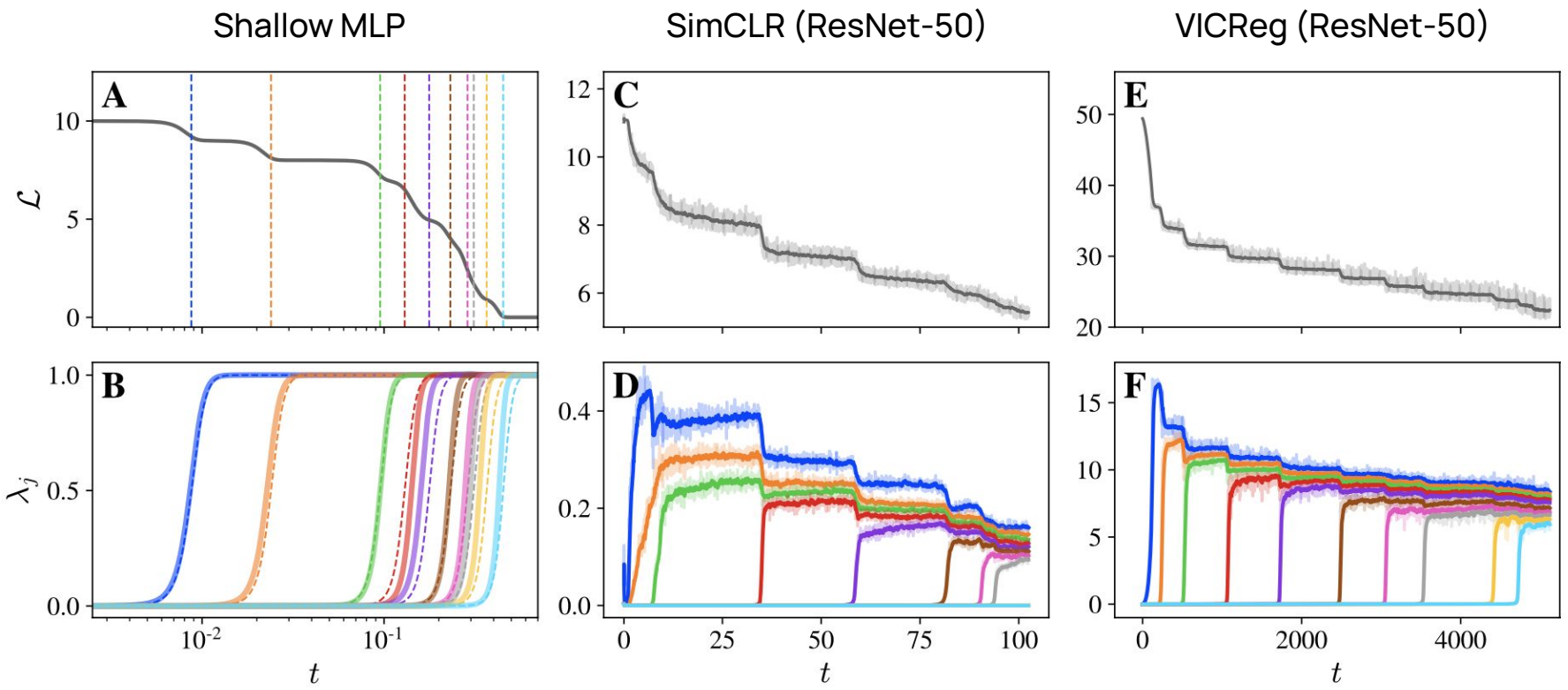
To kernelize, rephrase everything in terms of inner products $\boldsymbol{x}^T\boldsymbol{x}'$, then sub in $\mathrm{K}(\mathrm{x},\mathrm{x}') \equiv \boldsymbol{x}^T\boldsymbol{x}'$.

End of theory. Let’s check against ResNets.

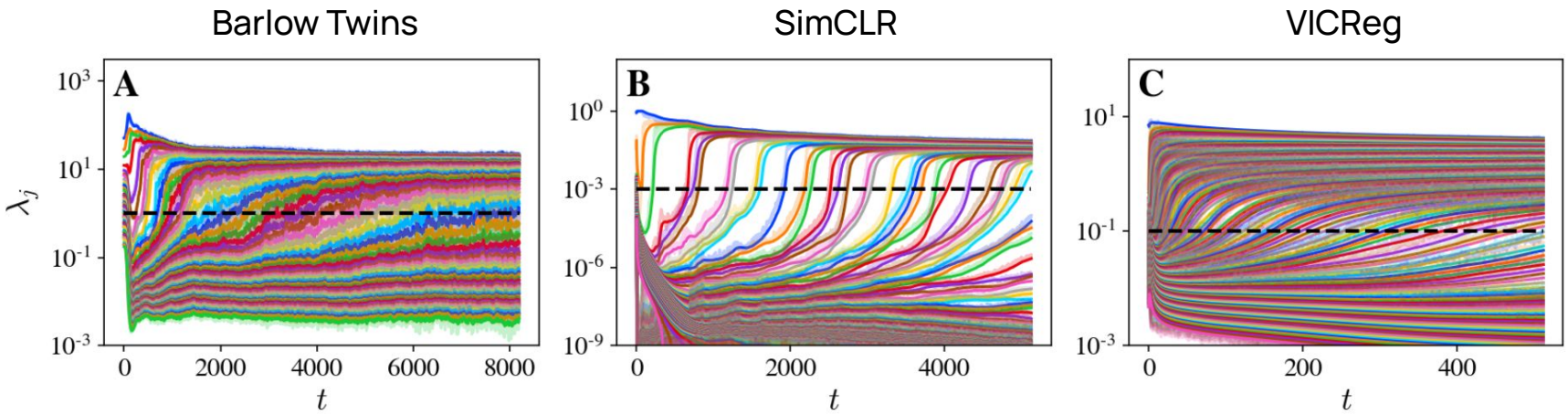
ResNets with small init show stepwise learning on STL-10.



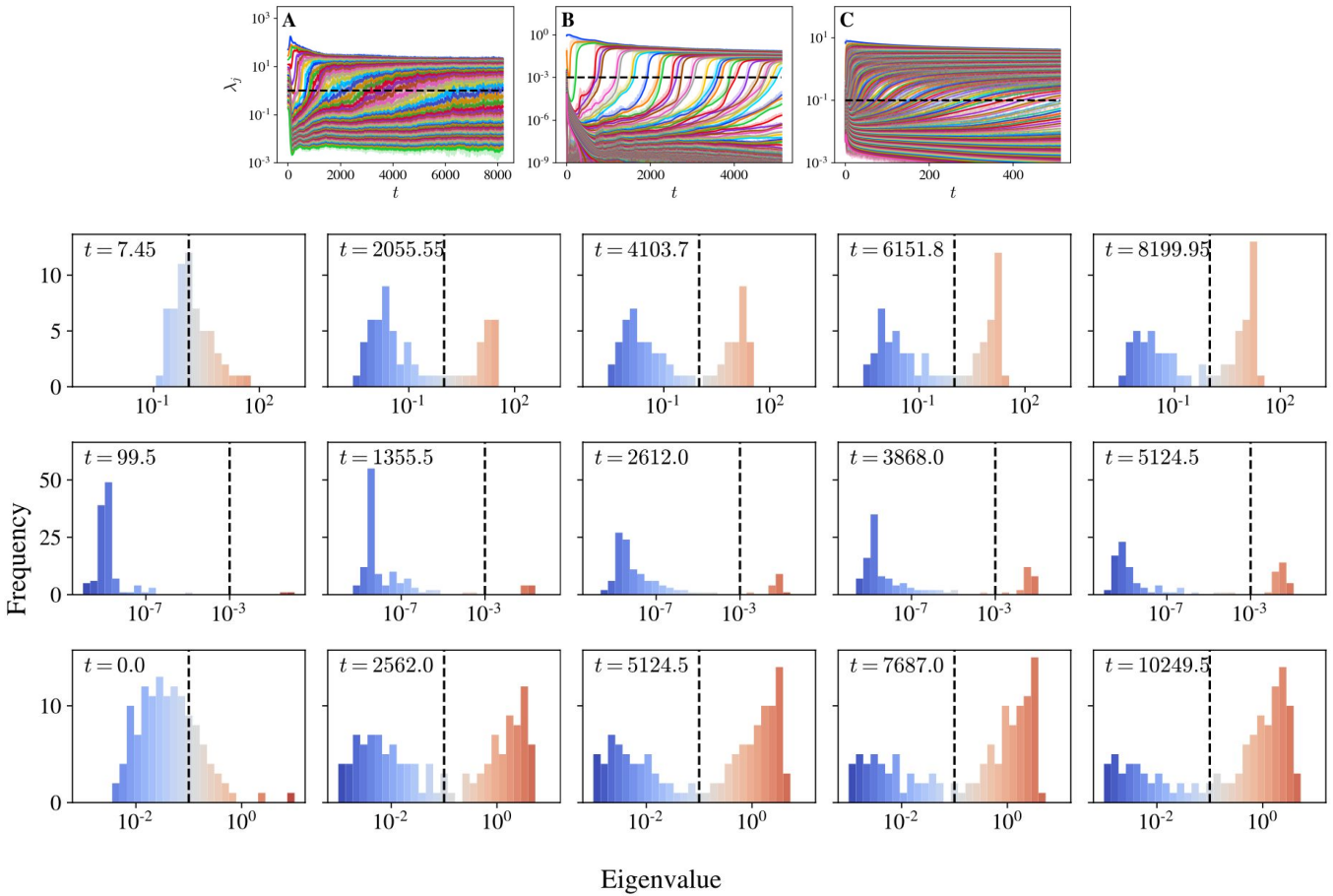
ResNets with small init show stepwise learning on STL-10 (cont.).



…and you can still see the stepwise behavior with regular init!



…and you can still see the stepwise behavior with regular init!



Future directions

Training SSL is famously slow. Can we speed it up?

Interpretability of top eigendirections?

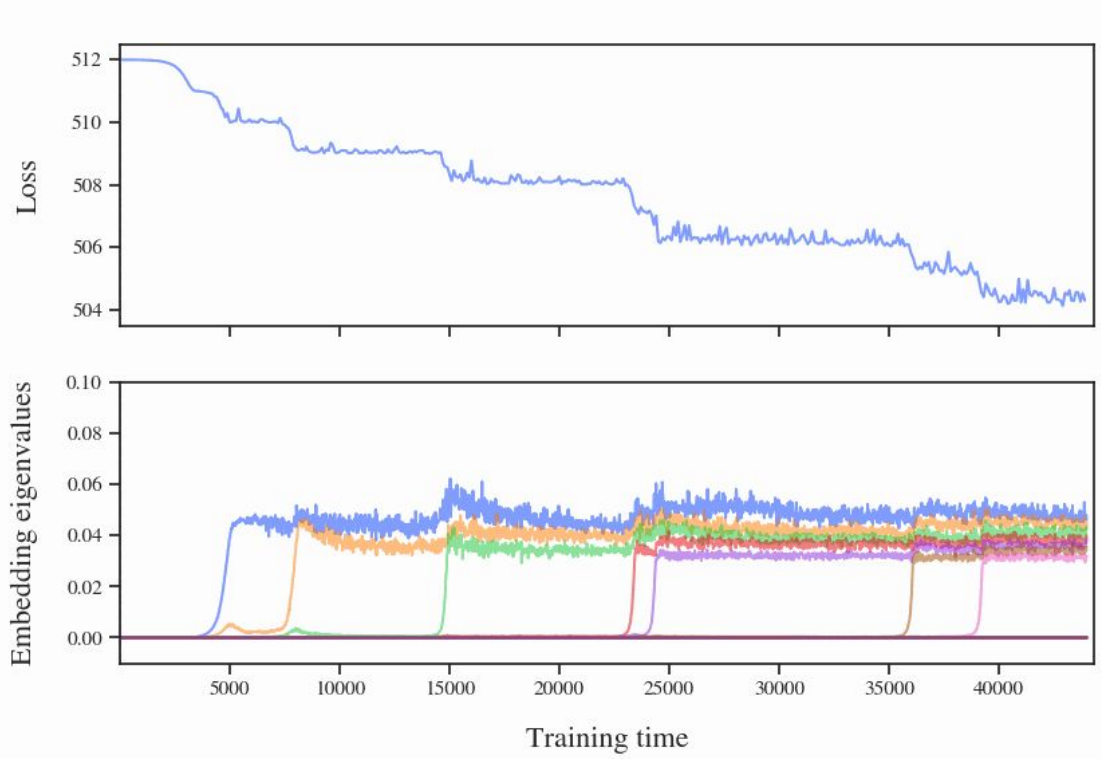
Stepwise learning in other settings? LLMs?

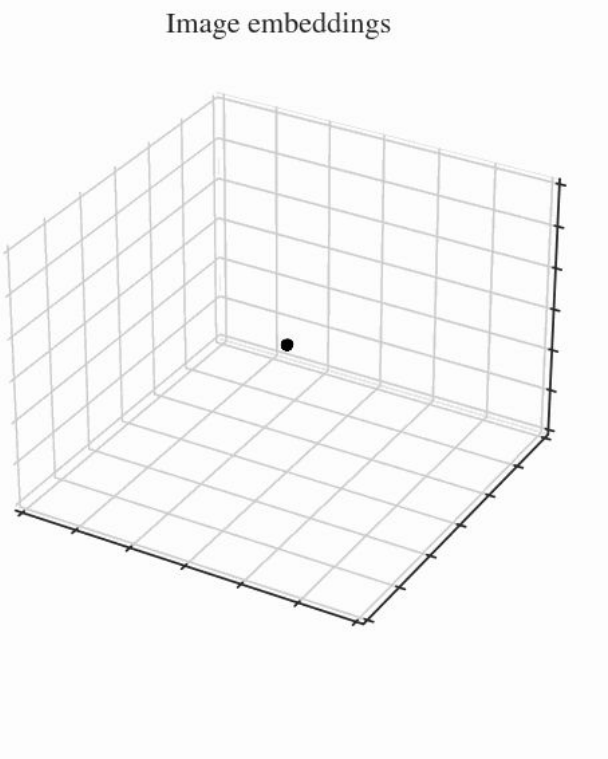
Broader lessons

1) Let's study (self/un)-supervised learning

2) It's worth collaborating with experimentalists

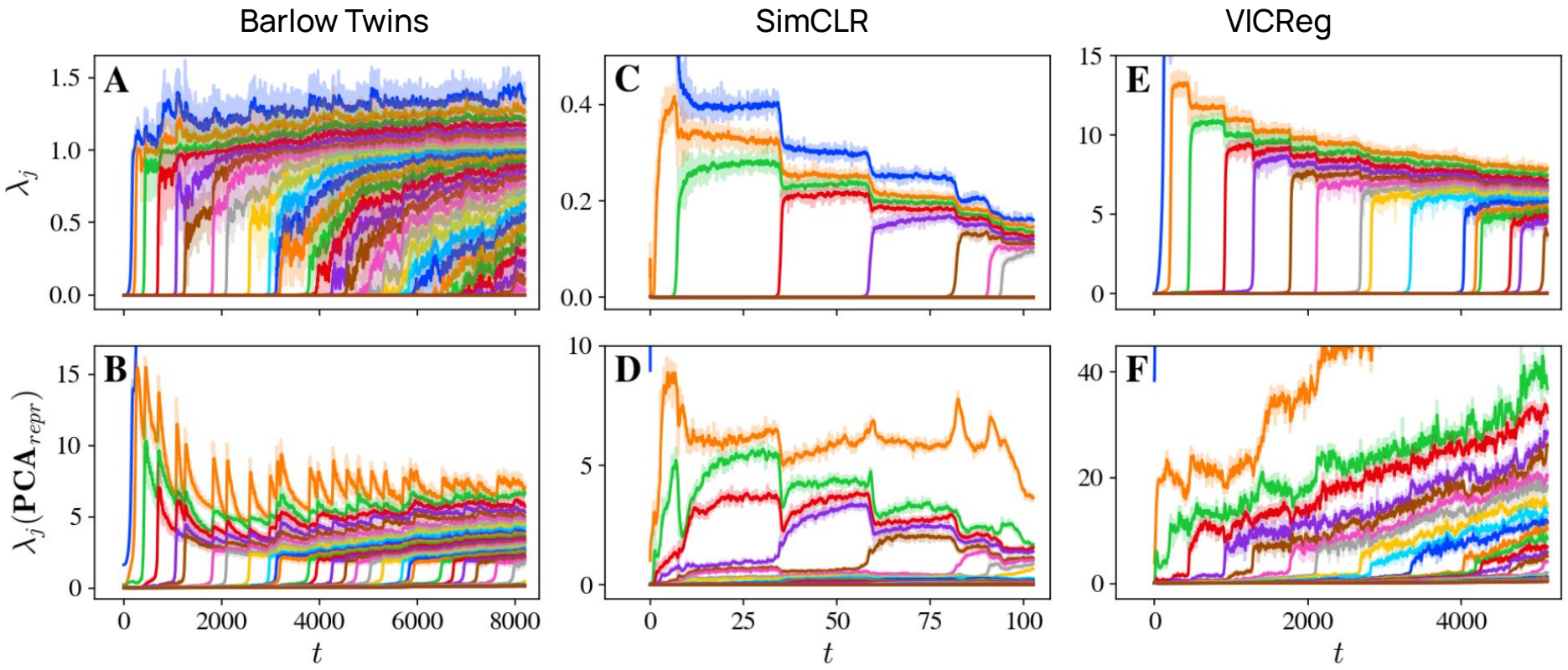
Summary: SOTA self-supervised systems show striking steps ('specially starting small)





(backups)

...and you can see it in *hidden representations* too!



We can mostly predict embeddings from the final NTK.

Procedure:

- Train a ResNet on full STL-10 dataset from small init.
- Measure eNTK after training on 1k positive pairs.
- Compute predicted embeddings from the eNTK.
- Compute inner product of true + predicted embedding subspaces.

	BT	SimCLR	VICReg	(random subspaces)
$\mathrm{a}(\boldsymbol{P}_{\mathrm{exp}}, \boldsymbol{P}_{\mathrm{th}})$	0.615	0.517	0.592	0.025

Not perfect, but pretty good agreement!