Understanding self-supervised Learning Dynamics without Contrastive Pairs



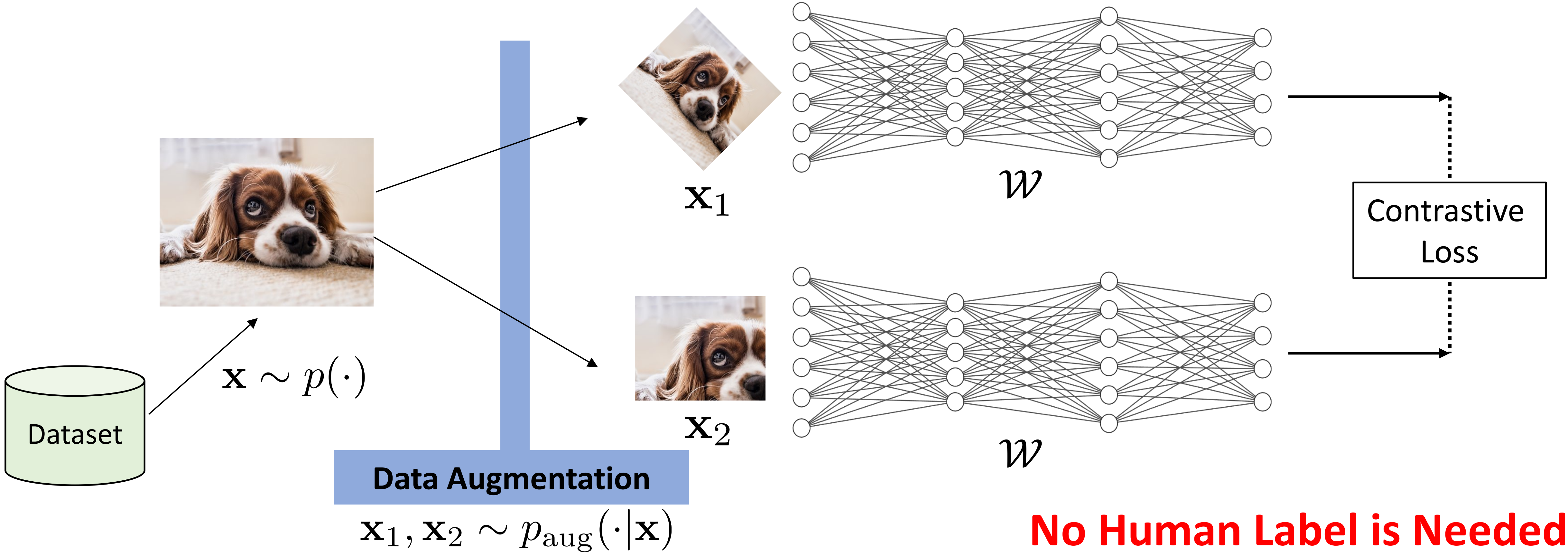
Yuandong Tian[1]   Xinlei Chen[1]   Surya Ganguli[1,2]

[1] Facebook AI Research   [2] Stanford University

Code: https://github.com/facebookresearch/luckmatter/ssl

facebook Artificial Intelligence

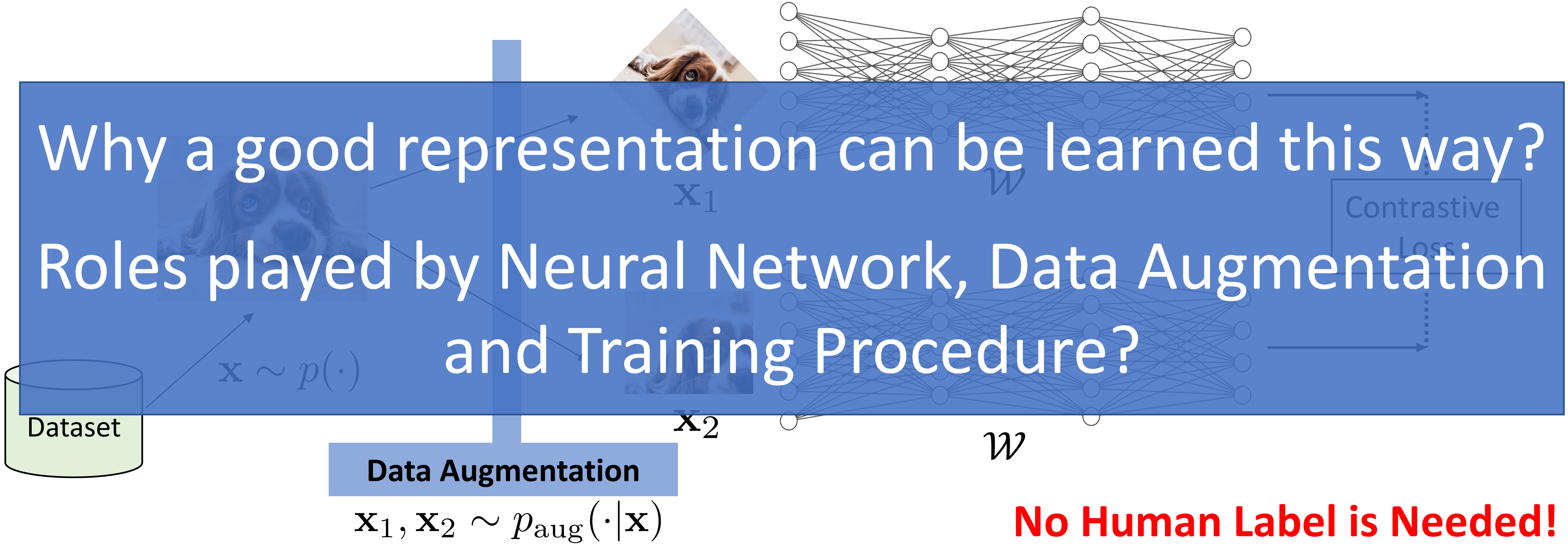
Self-supervised Learning (SimCLR)



No Human Label is Needed!

SimCLR: *[T. Chen, A Simple Framework for Contrastive Learning of Visual Representations, ICML 2020]*

Self-supervised Learning (SimCLR)

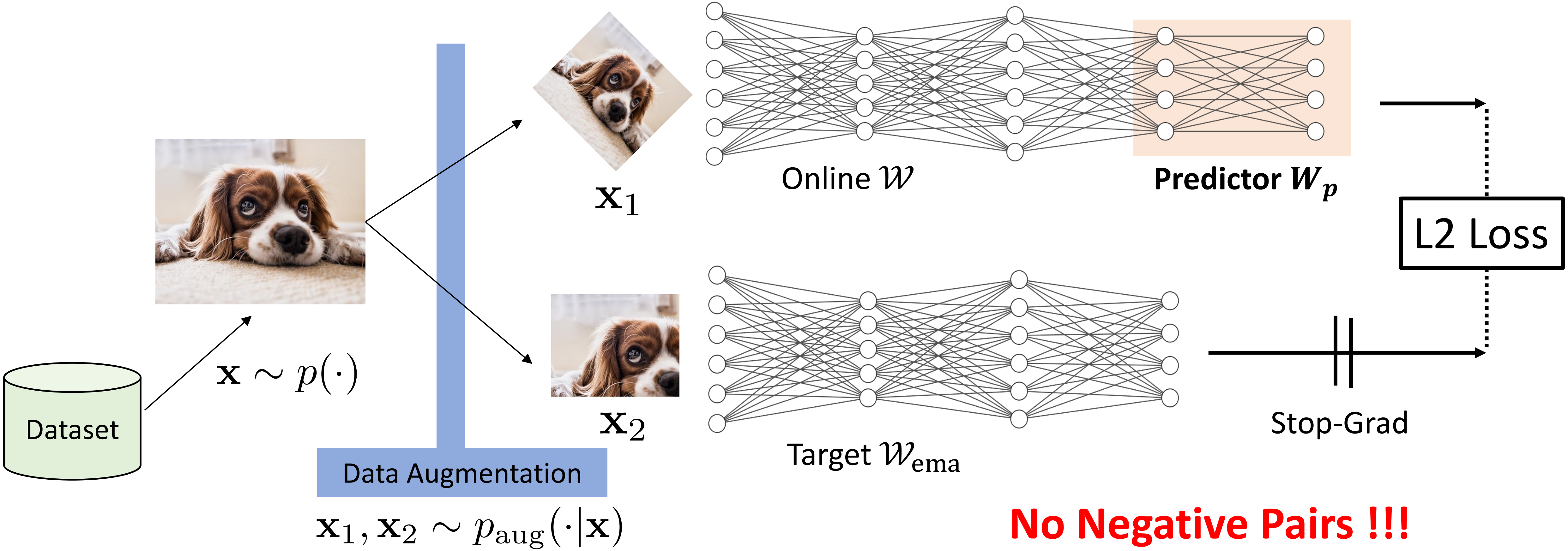


Why a good representation can be learned this way?

Roles played by Neural Network, Data Augmentation and Training Procedure?

No Human Label is Needed!

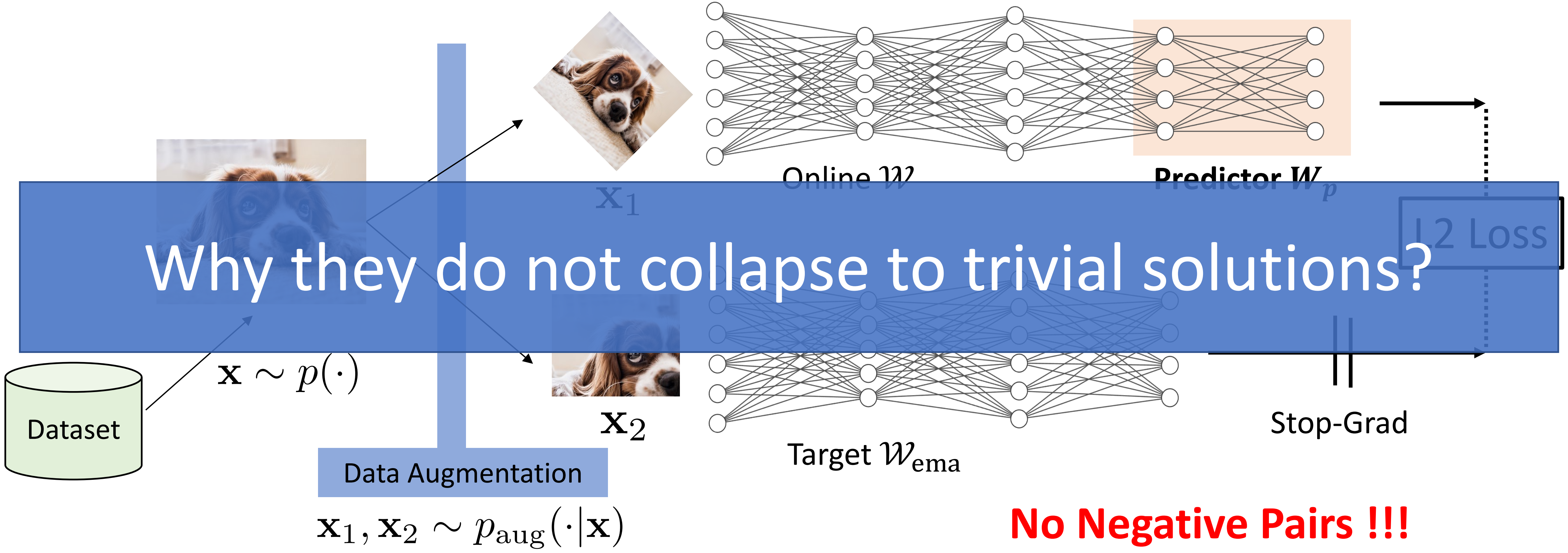
Non-contrastive SSL (BYOL/SimSiam)?



BYOL: *[J. Grill, Bootstrap your own latent: A new approach to self-supervised Learning, NeurIPS 2020]*

SimSiam: *[X. Chen and K. He, Exploring Simple Siamese Representation Learning, CVPR 2021]*

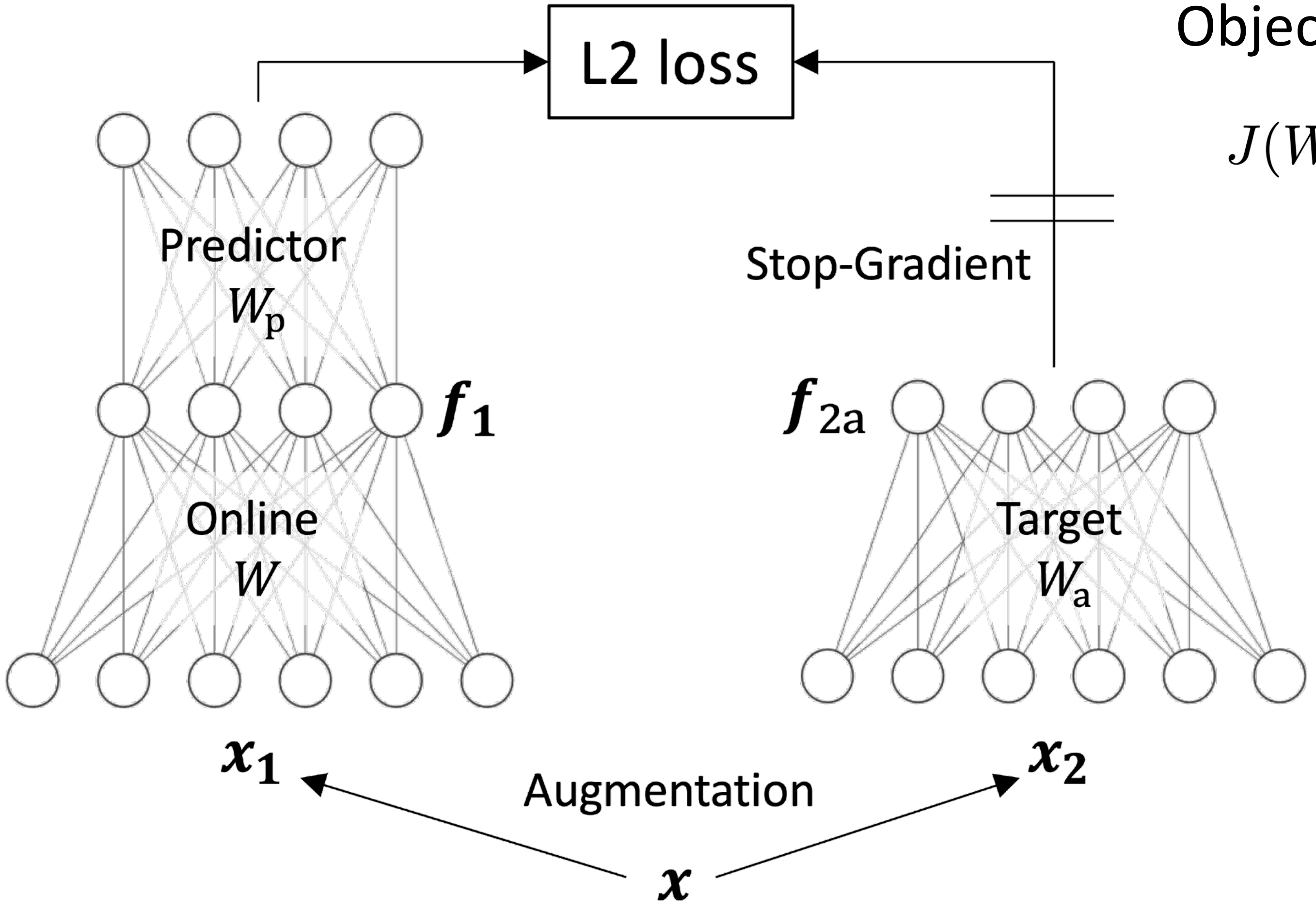
Non-contrastive SSL (BYOL/SimSiam)?



BYOL: *[J. Grill, Bootstrap your own latent: A new approach to self-supervised Learning, NeurIPS 2020]*

SimSiam: *[X. Chen and K. He, Exploring Simple Siamese Representation Learning, CVPR 2021]*

A simple model



Objective:

$$J(W, W_p) := \frac{1}{2}\mathbb{E}_{\boldsymbol{x}_1,\boldsymbol{x}_2}\left[\|W_p\boldsymbol{f}_1 - \mathrm{StopGrad}(\boldsymbol{f}_{2\mathrm{a}})\|_2^2\right]$$

Linear online network W

Linear target network W_a

Linear predictor W_p

Learning Dynamics

$$\bar{\boldsymbol{x}}(\boldsymbol{x}) := \mathbb{E}_{\boldsymbol{x}' \sim p_{\text{aug}}(\cdot|\boldsymbol{x})}[\boldsymbol{x}']$$
$$X = \mathbb{E}[\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^\intercal]$$
$$X' = \mathbb{E}_{\boldsymbol{x}}[\mathbb{V}_{\boldsymbol{x}'|\boldsymbol{x}}[\boldsymbol{x}']]$$

Lemma 1. *BYOL learning dynamics following Eqn. 1:*

$$\dot{W}_p = \alpha_p(-W_p W(X + X') + W_a X) W^\intercal - \eta W_p$$
$$\dot{W} = W_p^\intercal(-W_p W(X + X') + W_a X) - \eta W$$
$$\dot{W}_a = \beta(-W_a + W)$$

Hyper-parameter	Description
α_p	Relative learning rate of the predictor
η	Weight decay
β	The rate of Exponential Moving Average (EMA)

Stop-Gradient do not work

Theorem 2: No Stop-Gradient doesn't work ($W \to 0$)

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathrm{vec}(W) = -\left[X' \otimes (W_p^\intercal W_p + I) + X \otimes \tilde{W}_p^\intercal \tilde{W}_p\right] \mathrm{vec}(W)$$

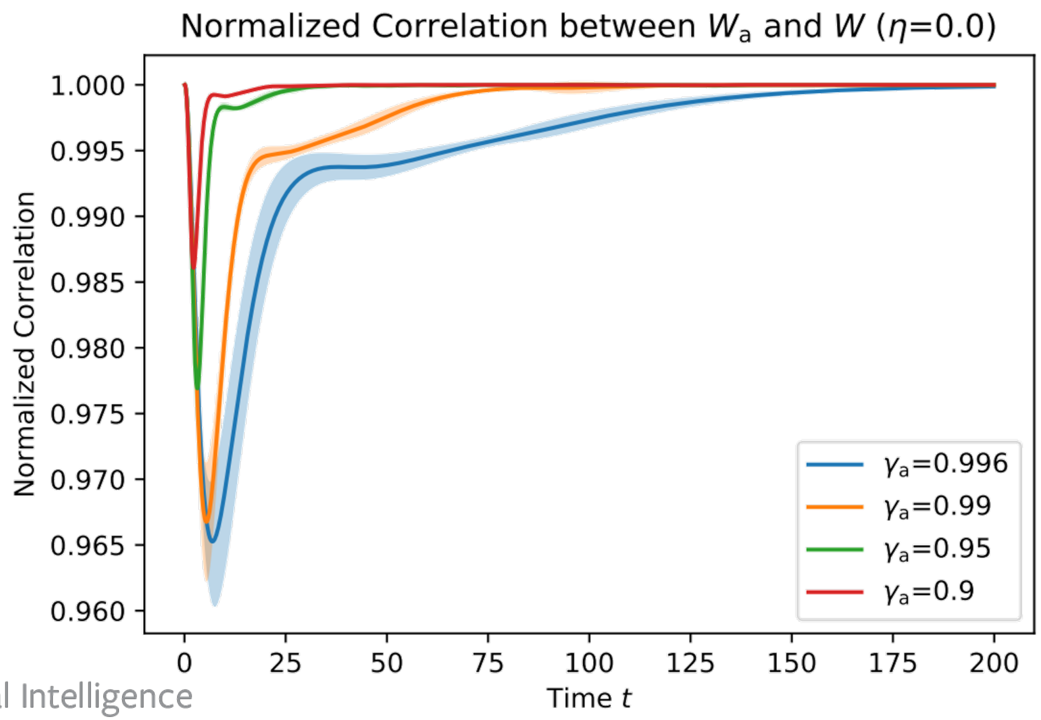
PSD matrix

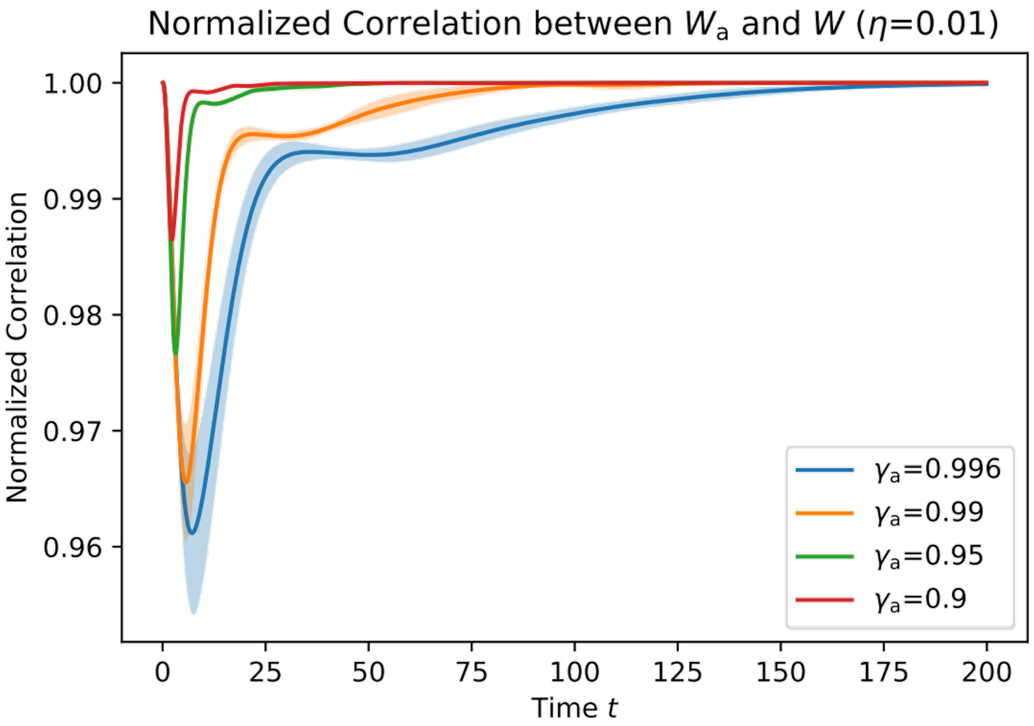
Here $\widetilde{W_p} := W_p - I$

Assumptions

Assumption 1 (Isotropic Data and Augmentation): $X = I$ and $X' = \sigma^2 I$

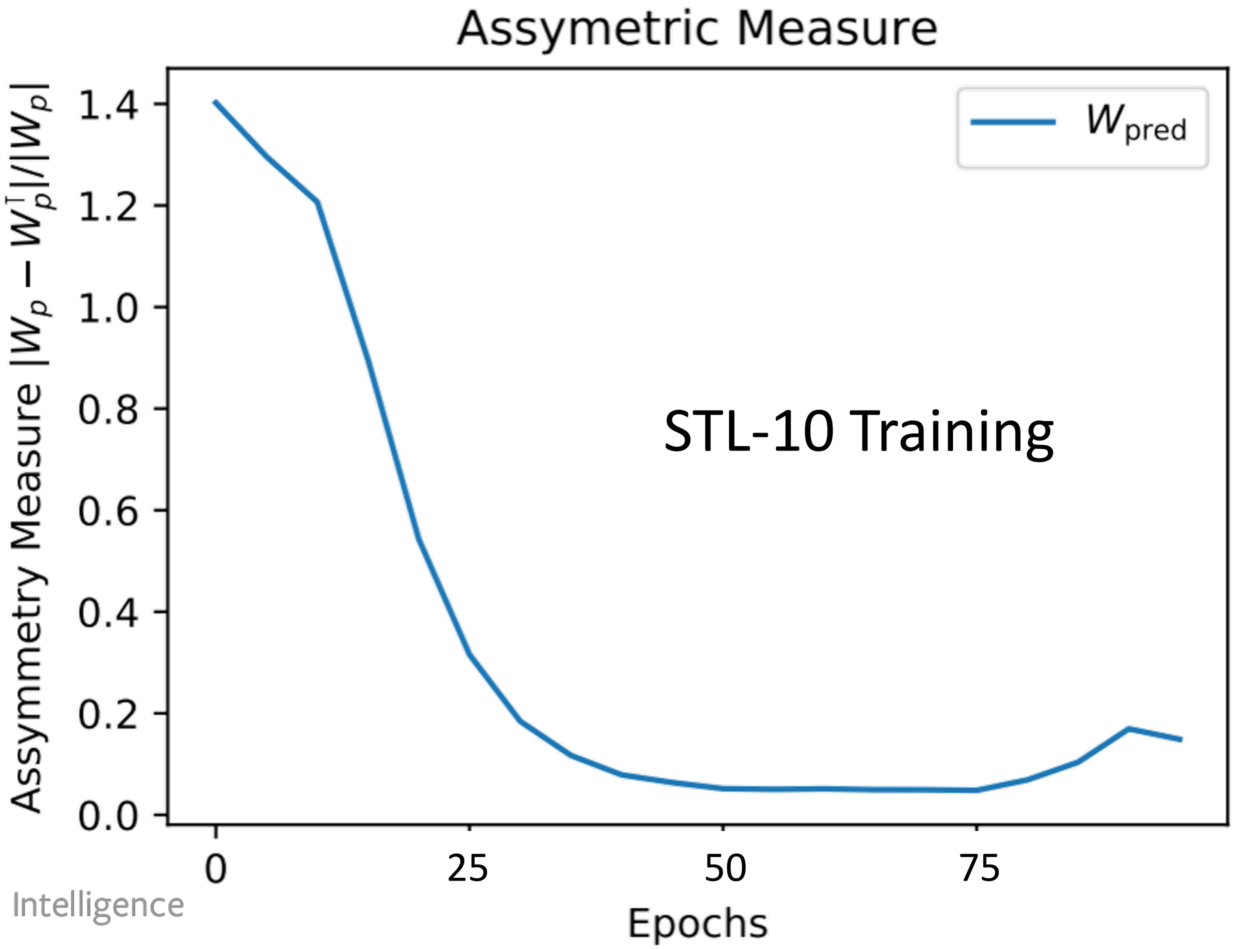
Assumption 2: the EMA weight $W_a(t) = \tau(t)W(t)$ is a linear function of $W(t)$





Symmetrization of the dynamics

Assumption 3 (Symmetric predictor W_p): $W_p(t) = W_p^T(t)$



W_p becomes more and more **symmetric** over training

The effect of Symmetrized Predictor W_p

	No predictor bias		With predictor bias	
	sym W_p	regular W_p	sym W_p	regular W_p
One-layer linear predictor				
EMA	75.09 ± 0.48	74.51 ± 0.47	74.52 ± 0.29	74.16 ± 0.33
no EMA	**36.62 ± 1.85**	72.85 ± 0.16	**36.04 ± 2.74**	72.13 ± 0.53
Two-layer predictor with BatchNorm and ReLU				
EMA	71.58 ± 6.46	78.85 ± 0.25	77.64 ± 0.41	78.53 ± 0.34
no EMA	**35.59 ± 2.10**	65.98 ± 0.71	**41.92 ± 4.25**	65.59 ± 0.66

Symmetric W_p affects the performance a lot!

Symmetrized Dynamics

Define **anti-commutator** $\{A, B\} := AB + BA$:

$$\begin{aligned} \dot{W}_p &= -\frac{\alpha_p}{2}(1+\sigma^2)\{W_p, F\} + \alpha_p \tau F - \eta W_p \\ \dot{F} &= -(1+\sigma^2)\{W_p^2, F\} + \tau\{W_p, F\} - 2\eta F \end{aligned}$$

Here $F := \mathrm{E}[ff^T] = WXW^T$

is the correlation matrix of the input of the predictor.

Eigenspace Alignment

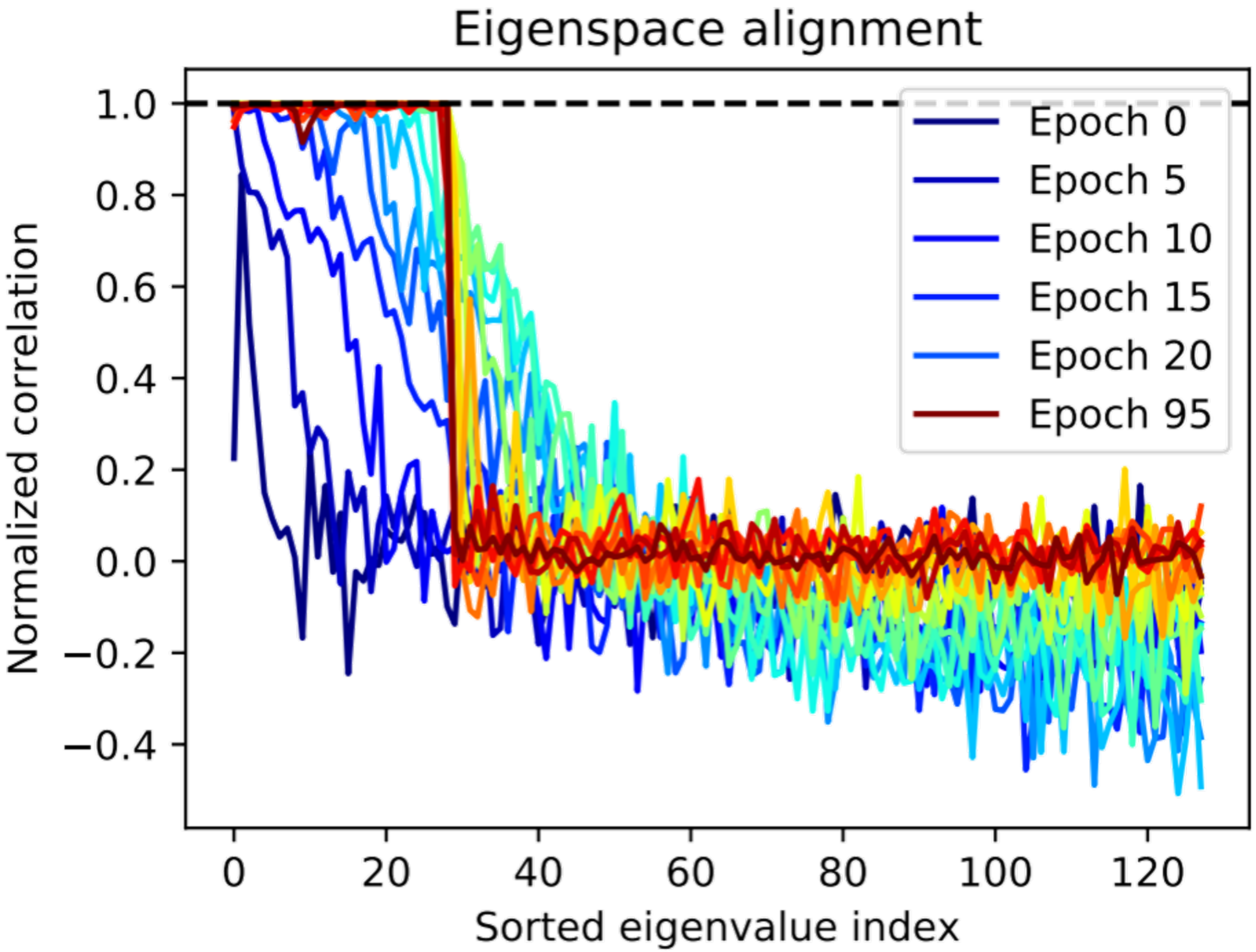
Theorem 3: Under certain conditions,

$$FW_p - W_pF \to 0 \text{ when } t \to +\infty$$

and thus the eigenspace of W_p and F gradually aligns.

Empirical Result says the same

STL-10 Training



Decoupled dynamics

When eigenspace aligns, the dynamics becomes decoupled:

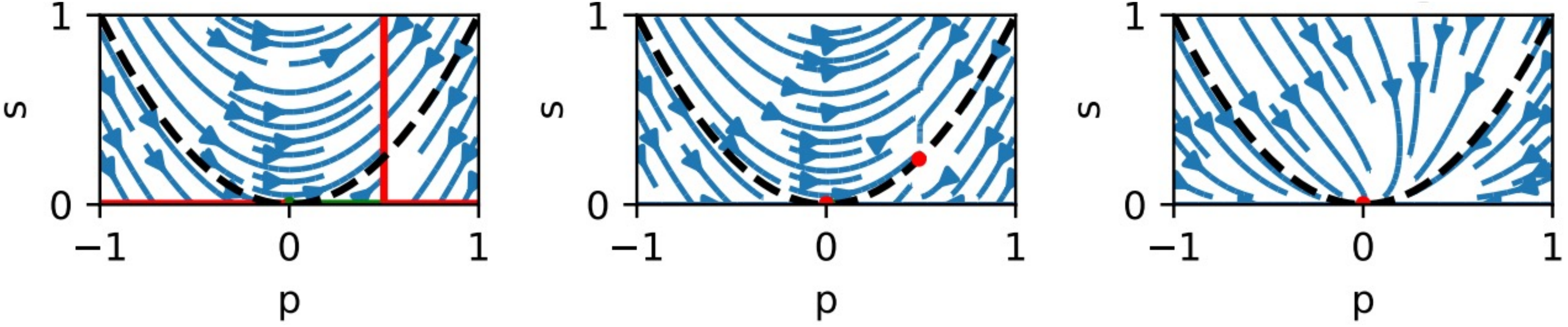
$$
\begin{aligned}
\dot{p}_j &= \alpha_p s_j \left[\tau - (1+\sigma^2)p_j\right] - \eta p_j \\
\dot{s}_j &= 2p_j s_j \left[\tau - (1+\sigma^2)p_j\right] - 2\eta s_j \\
s_j \dot{\tau} &= \beta(1-\tau)s_j - \tau \dot{s}_j/2.
\end{aligned}
$$

Where p_j and s_j are eigenvalues of W_p and F

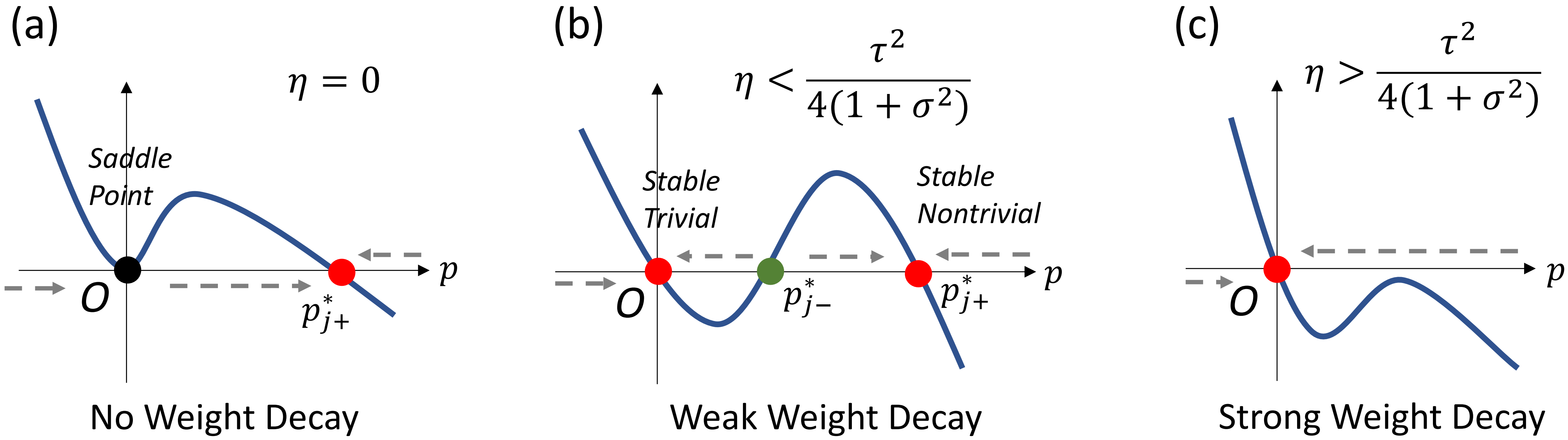
Invariance holds: $s_j(t) = \alpha_p^{-1} p_j^2(t) + e^{-2\eta t} c_j$

State Space Dynamics (Phase Diagram)

No weight decay ($\eta = 0$) | Weak weight decay ($\eta = 0.01$) | Strong weight decay ($\eta = 1$)



Why BYOL doesn't collapse?

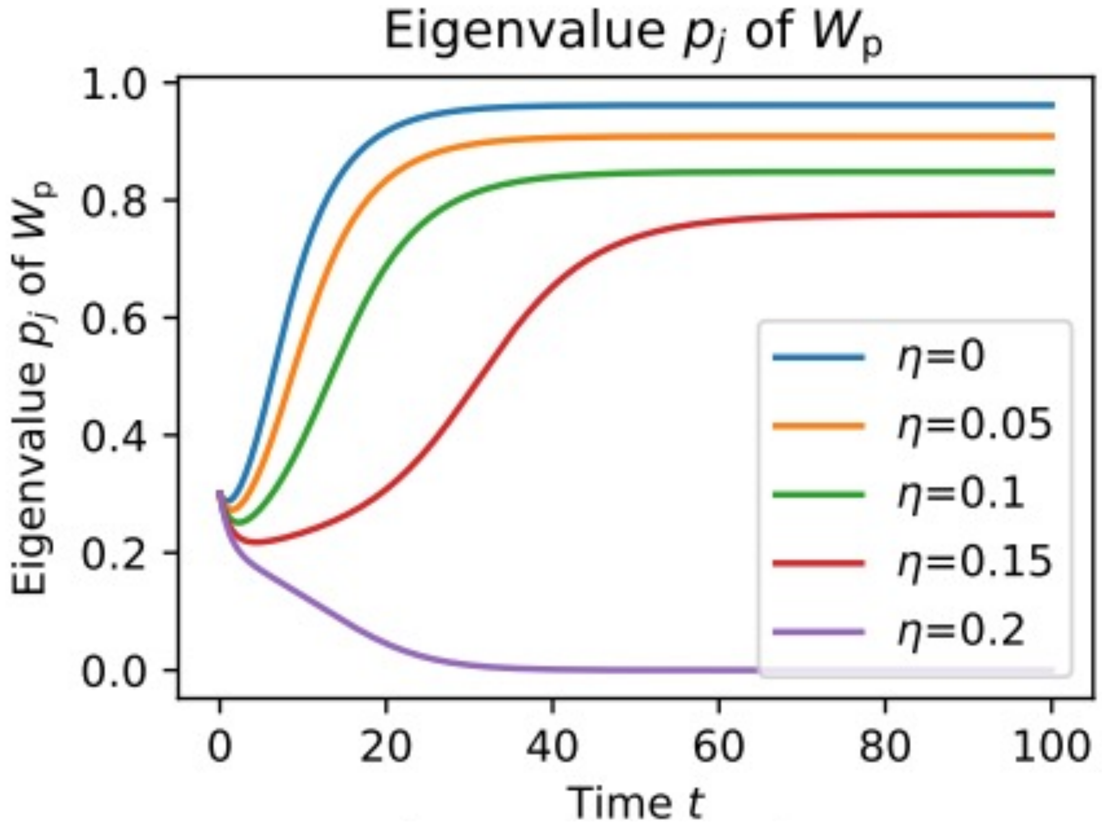


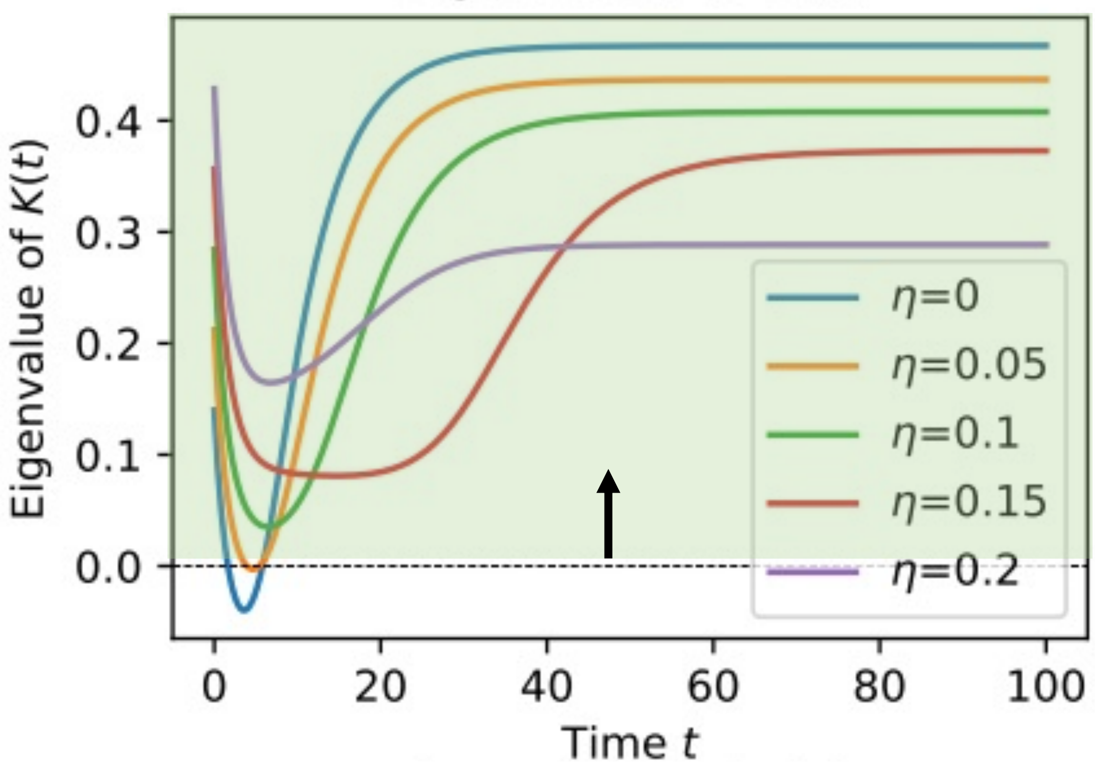
The Benefit of Weight Decay

Let $\Delta_j := p_j[\tau - (1+\sigma^2)p_j] - \eta$

Eigenspace alignment condition

$$\Delta_j < \frac{1}{2}[\alpha_p(1+\sigma^2)s_j + \eta]$$



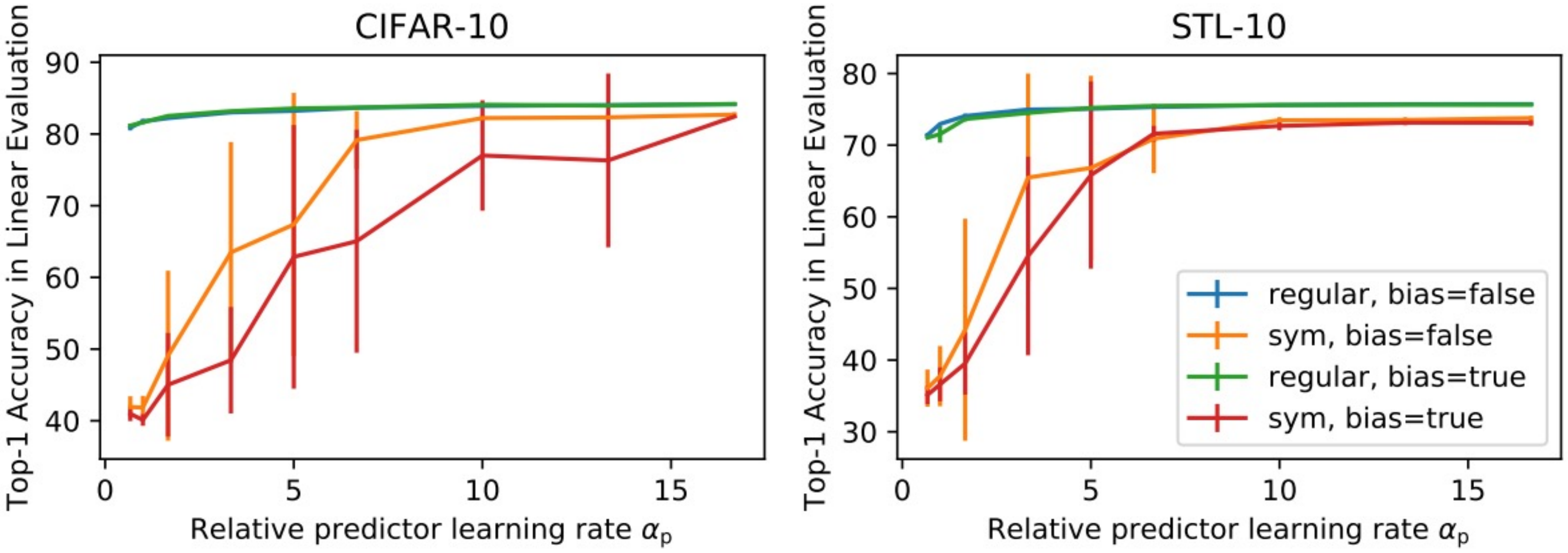


Higher weight decay leads to better satisfaction of alignment condition!

Relative learning rate of the predictor $\boldsymbol{\alpha_p}$

Positive ☺

1. Large α_p shrinks the size of trivial basin
2. Relax the condition of eigenspace alignment



Negative ☹ With very large α_p, eigenvalue of F won't grow (and no feature learning)

Exponential Moving Average rate β

β large → $W_a(t)$ catches $W(t)$ faster

Positive ☺: Slower rate (small β) relaxes the condition of eigenspace alignment

τ needs to be small to satisfy **the eigenspace alignment condition**

$$p_j\tau - (1+\sigma^2)p_j^2 < \frac{\alpha_p}{2}(1+\sigma^2)s_j + \frac{3}{2}\eta$$

first order — $p_j\tau$

second order — $(1+\sigma^2)p_j^2$

$s_j \sim p_j^2$ **second order** — $(1+\sigma^2)s_j$

Negative ☹: Slower rate makes the training slow and expands the size of trivial basin

DirectPred

- Directly setting W_p rather than relying on gradient descent update.

  1. Estimate $\hat{F} = \rho\hat{F} + (1-\rho)E[\boldsymbol{f}\boldsymbol{f}^T]$
  2. Eigen-decompose $\hat{F} = \hat{U}\Lambda_F\hat{U}^T$, $\Lambda_F = \text{diag}\,[s_1, s_2, \ldots, s_d]$
  3. Set W_p following the invariance:

$$p_j = \sqrt{s_j} + \epsilon \max_j s_j, \quad W_p = \hat{U}\text{diag}[p_j]\hat{U}^\intercal$$

Guaranteed Eigenspace Alignment ☺

Performance of DirectPred on STL-10/CIFAR-10

Downstream Classification Top-1	Number of epochs 100	300	500
STL-10			
DirectPred	$\mathbf{77.86 \pm 0.16}$	78.77 ± 0.97	78.86 ± 1.15
DirectPred (freq=5)	77.54 ± 0.11	$\mathbf{79.90 \pm 0.66}$	$\mathbf{80.28 \pm 0.62}$
SGD baseline	75.06 ± 0.52	75.25 ± 0.74	75.25 ± 0.74
CIFAR-10			
DirectPred	$\mathbf{85.21 \pm 0.23}$	$\mathbf{88.88 \pm 0.15}$	89.52 ± 0.04
DirectPred (freq=5)	84.93 ± 0.29	88.83 ± 0.10	$\mathbf{89.56 \pm 0.13}$
SGD baseline	84.49 ± 0.20	88.57 ± 0.15	89.33 ± 0.27

Performance of DirectPred on ImageNet

ImageNet performance (60 epoch)

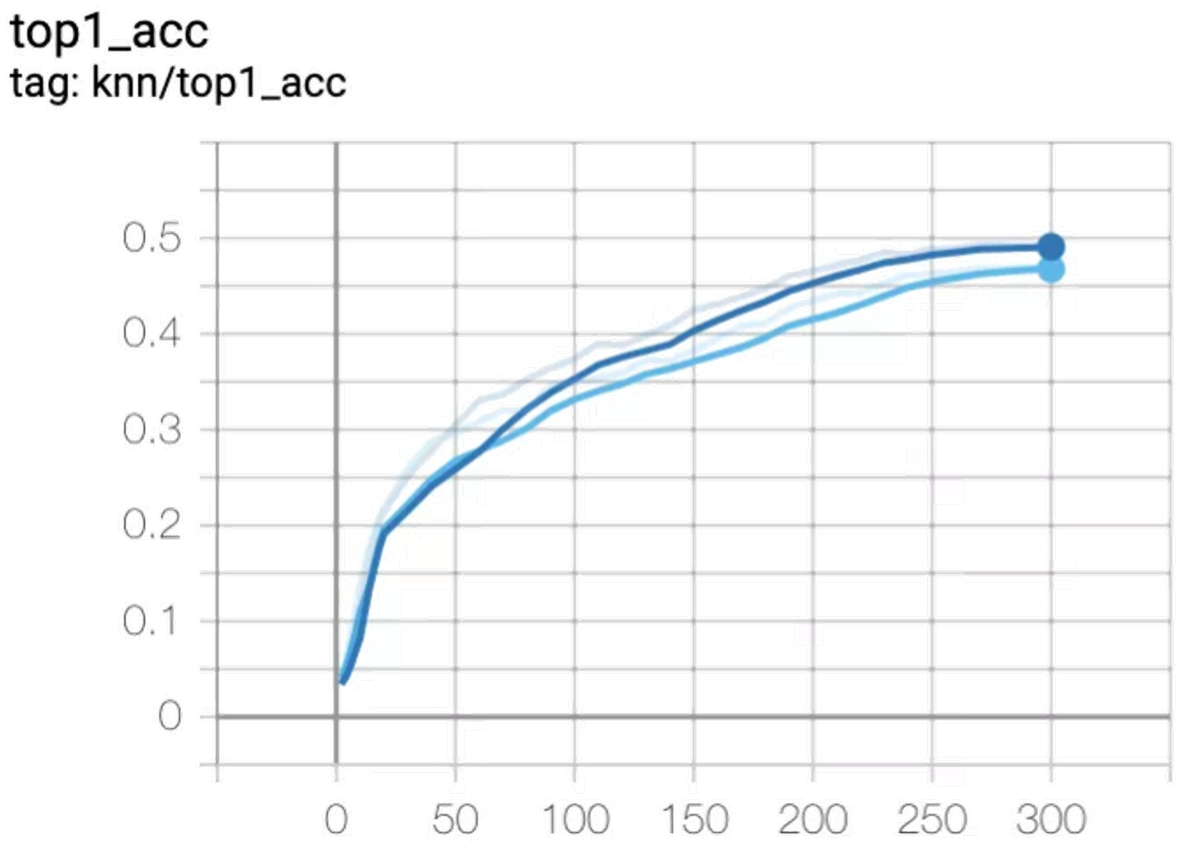
BYOL variants	Accuracy Top-1	Accuracy Top-5
2-layer predictor (default)	**64.7**	**85.8**
linear predictor	59.4	82.3
DirectPred	64.4	**85.8**

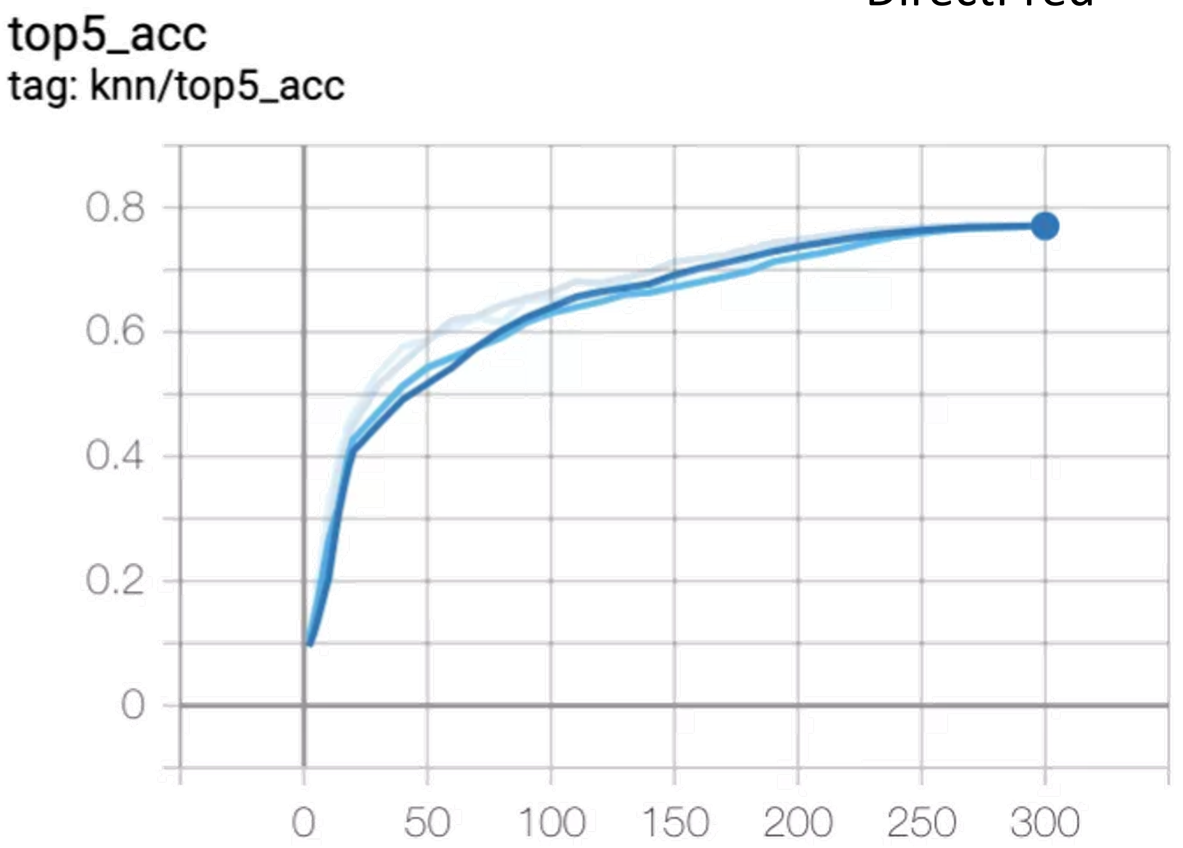
DirectPred using linear predictor is better than SGD with linear predictor, and is comparable with 2-layer predictor.

Performance of DirectPred on ImageNet

ImageNet performance (300 epoch)

BYOL
DirectPred





Thanks!