Understanding Self-Supervised Learning Dynamics without Contrastive Pairs

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

Abstract

While contrastive approaches of self-supervised learning (SSL) learn representations by minimizing the distance between two augmented views of the same data point (positive pairs) and maximizing views from different data points (negative pairs), recent *non-contrastive* SSL (e.g., BYOL and SimSiam) show remarkable performance *without* negative pairs, with an extra learnable predictor and a stop-gradient operation. A fundamental question arises: why do these methods not collapse into trivial representations? We answer this question via a simple theoretical study and propose a novel approach, **DirectPred**, that *directly* sets the linear predictor based on the statistics of its inputs, without gradient training. On ImageNet, it performs comparably with more complex two-layer non-linear predictors that employ BatchNorm and outperforms a linear predictor by 2.5% in 300-epoch training (and 5% in 60-epoch). **DirectPred** is motivated by our theoretical study of the nonlinear learning dynamics of non-contrastive SSL in simple linear networks. Our study yields conceptual insights into how non-contrastive SSL methods learn, how they avoid representational collapse, and how multiple factors, like predictor networks, stop-gradients, exponential moving averages, and weight decay all come into play. Our simple theory recapitulates the results of real-world ablation studies in both STL-10 and ImageNet. Code is released[1].

1. Introduction

Self-supervised learning (SSL) has emerged as a powerful method for learning useful representations without requiring expensive target labels (Devlin et al., 2018). Many state-of-the-art SSL methods in computer vision employ the principle of contrastive learning (Oord et al., 2018; Tian et al., 2019; He et al., 2020; Chen et al., 2020a; Bachman et al., 2019) whereby the hidden representations of two augmented views of the same object (positive pairs) are brought closer together, while those of different objects (negative pairs) are encouraged to be further apart. Minimizing differences between positive pairs encourages modeling invariances, while contrasting negative pairs is thought to be required to prevent representational collapse (i.e., mapping all data to the same representation).

However, some recent SSL work, notably BYOL (Grill et al., 2020) and SimSiam (Chen & He, 2020), have shown the remarkable capacity to learn powerful representations using only positive pairs, *without* ever contrasting negative pairs. These methods employ a dual pair of Siamese networks (Bromley et al., 1994) (Fig. 1): the representation of two views are trained to match, one obtained by the composition of an online and predictor network, and the other by a target network. The target network is *not* trained via gradient descent; and either employs a direct copy of the online network (e.g., SimSiam (Chen & He, 2020)), or a momentum encoder that slowly follows the online network in a delayed fashion through an exponential moving average (EMA) (e.g., MoCo (He et al., 2020; Chen et al., 2020b) and BYOL (Grill et al., 2020)). Compared to contrastive learning, these non-contrastive SSL methods do not require large batch size (e.g., 4096 in SimCLR (Chen et al., 2020a)) or memory queue (e.g., MoCo (He et al., 2020; Chen et al., 2020b)) to provide negative pairs. Therefore, they are generally more efficient and conceptually simple while maintaining state-of-the-art performance.

Since the entire procedure in non-contrastive SSL encourages the online+predictor network and the target network to become similar to each other, this overall scheme raises several fundamental unsolved theoretical questions. Why/how does it avoid collapsed representations? What is the nature of the learned representations? How do multiple design choices and hyperparameters interact nonlinearly in the learning dynamics? While there are interesting theoretical studies of contrastive SSL (Arora et al., 2019; Lee et al., 2020; Tosh et al., 2020), any theoretical understanding of the nonlinear learning dynamics of non-contrastive

[1]Facebook AI Research [2]Stanford University. Correspondence to: Yuandong Tian <yuandong@fb.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

[1]https://github.com/facebookresearch/luckmatters/tree/master/ssl

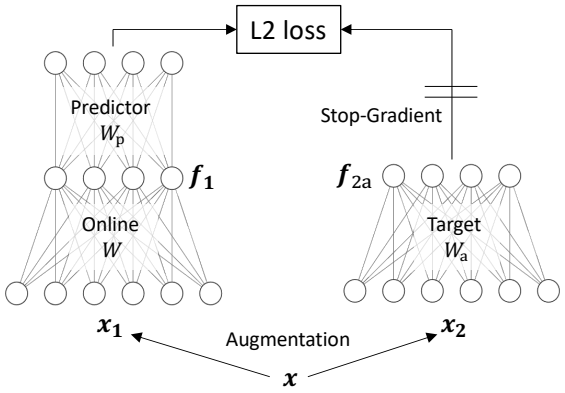

Figure 1. Two-layer setting with a linear, bias-free predictor.

SSL remains open.

In this paper, we make a first attempt to analyze the behavior of non-contrastive SSL training and the empirical effects of multiple hyperparameters, including **(1)** Exponential Moving Average (EMA) or momentum encoder, **(2)** Higher relative learning rate (α_p) of the predictor, and **(3)** Weight decay η. We explain all these empirical findings with an exceedingly simple theory based on analyzing the nonlinear learning dynamics of simple linear networks. Note that deep linear networks have provided a useful tractable theoretical model of nonconvex loss landscapes (Kawaguchi, 2016; Du & Hu, 2019; Laurent & Brecht, 2018) and nonlinear learning dynamics (Saxe et al., 2013; 2019; Lampinen & Ganguli, 2018; Arora et al., 2018) in these landscapes, yielding insights like dynamical isometry (Saxe et al., 2013; Pennington et al., 2017; 2018) that lead to improved training of nonlinear deep networks. Despite the simplicity of our theory, it can still predict how various hyperparameter choices affect performance in an extensive set of real-world ablation studies. Moreover, the simplicity also enables us to provide conceptual and analytic insights into *why* performance patterns vary the way they do. Specifically, our theory accounts for the following diverse empirical findings:

Essential part of non-contrastive SSL. The existence of the predictor and stop-gradient is absolutely essential. Removing either of them leads to representational collapse in BYOL and SimSiam.

EMA. While the original BYOL needs EMA to work, they later confirmed that EMA is not necessary (i.e., the online and target networks can be identical) if a higher α_p is used. This is also confirmed with SimSiam, as long as the predictor is updated more often or has larger learning rate (or larger α_p). However, the performance is slightly lower.

Predictor Optimality and Relative learning rate α_p. Both BYOL and SimSiam suggest that the predictor should always be optimal, in the sense of always achieving minimal ℓ_2 error in predicting the target network's outputs from the online network's outputs. This optimality conjecture was motivated by observed superior performance when the predictor had large learning rates and/or was allowed more frequent updates than the rest of the network. However (Chen & He, 2020) also showed that if the predictor is updated too often, then performance drops, which questions the importance of an always optimal predictor as a key requirement for learning good representations.

	Plug-in frequency (every N minibatches)			
	1	2	3	5
EMA	40.67 ± 0.50	35.29 ± 2.49	34.60 ± 0.98	35.63 ± 2.66
no EMA	39.45 ± 1.26	34.01 ± 1.54	34.58 ± 2.93	32.22 ± 2.94

Table 1. Simply plugging in the "optimal solution" to the linear predictor shows poor performance after 100 BYOL epochs (Top-1 accuracy in STL-10 (Coates et al., 2011) downstream classification task). The optimal solution is obtained by solving (with regularization) $W_p\mathbb{E}\left[\boldsymbol{f}\boldsymbol{f}^\intercal\right] = \frac{1}{2}(\mathbb{E}\left[\boldsymbol{f}_\mathrm{a}\boldsymbol{f}^\intercal\right] + \mathbb{E}\left[\boldsymbol{f}\boldsymbol{f}_\mathrm{a}^\intercal\right])$, in which the two expectations is estimated with exponential moving average. In comparison, with gradient descent, BYOL with a single linear layer predictor can reach 74%-75% Top-1 in STL-10 after 100 epochs. Unless explicitly stated, in all our experiments, we use ResNet-18 (He et al., 2016) as the backbone network for CIFAR-10/STL-10 experiments and SGD as the optimizer with learning rate $\alpha = 0.03$, momentum 0.9, weight decay $\bar{\eta} = 0.0004$ and EMA parameter $\gamma_\mathrm{a} = 0.996$. Each setting is repeated 5 times.

Weight Decay. Table 15 in BYOL (Grill et al., 2020) indicates that no weight decay may lead to unstable results. A recent blogpost (Fetterman & Albrecht, 2020) also mentions using weight decay leads to stable learning in BYOL.

Finally, motivated by our theoretical analysis, we propose a new method **DirectPred** that directly sets the predictor weights based on principal components analysis of the predictor's input, thereby avoiding complicated predictor dynamics and initialization issues. We show that this simple **DirectPred** method nevertheless yields comparable performance in CIFAR-10 and outperforms gradient training of the linear predictor by $+5\%$ Top-1 accuracy in linear evaluation protocol on both STL-10 and ImageNet (60 epochs). On the standard ImageNet benchmark (300 epochs), **DirectPred** achieves $72.4\%/91.0\%$ Top-1/Top-5, 2.5% higher than BYOL with linear predictor ($69.9\%/89.6\%$) and comparable with default BYOL setting with 2-layer predictor ($72.5\%/90.8\%$).

2. Two-layer linear model

To obtain analytic and conceptual insights into non-contrastive SSL we analyze a simple, *bias-free* linear BYOL model where the online, target and predictor networks are specified by the weight matrices $W \in \mathbb{R}^{n_2\times n_1}$, $W_p \in \mathbb{R}^{n_2\times n_2}$ and $W_\mathrm{a} \in \mathbb{R}^{n_2\times n_1}$ respectively (Fig. 1).

Let $\boldsymbol{x} \in \mathbb{R}^{n_1}$ be a data point drawn from the data distribution $p(\boldsymbol{x})$ and let $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ be two augmented views of $\boldsymbol{x}$: $\boldsymbol{x}_1, \boldsymbol{x}_2 \sim p_{\rm aug}(\cdot|\boldsymbol{x})$ where $p_{\rm aug}(\cdot|\boldsymbol{x})$ is the augmentation distribution. In practice such data augmentations correspond to random crops, blurs or color distortions of images (Chen et al., 2020a). Let $\boldsymbol{f}_1 = W\boldsymbol{x}_1 \in \mathbb{R}^{n_2}$ be the online representation of view 1, and $\boldsymbol{f}_{2\rm a} = W_{\rm a}\boldsymbol{x}_2 \in \mathbb{R}^{n_2}$ be the target representation of view 2. In BYOL, the learning dynamics of W and W_p are obtained by minimizing

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

while the dynamics of W_a is obtained differently, via an exponential moving average (EMA) of W. We will analyze this combined dynamics for W, W_p and W_a, in the presence of additional weight decay, in the limit of large batch sizes and small discrete time learning rates. This limit can be well approximated by the gradient flow (see Supplementary Material (SM) for all derivations):

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

$$\dot{W}_p = \alpha_p\left(-W_pW(X + X') + W_{\rm a}X\right)W^\intercal - \eta W_p \tag{2}$$

$$\dot{W} = W_p^\intercal\left(-W_pW(X + X') + W_{\rm a}X\right) - \eta W \tag{3}$$

$$\dot{W}_{\rm a} = \beta(-W_{\rm a} + W) \tag{4}$$

Here, $X := \mathbb{E}\left[\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^\intercal\right]$ where $\bar{\boldsymbol{x}}(\boldsymbol{x}) := \mathbb{E}_{\boldsymbol{x}'\sim p_{\rm aug}(\cdot|\boldsymbol{x})}\left[\boldsymbol{x}'\right]$ is the average augmented view of a data point $\boldsymbol{x}$ and $X' := \mathbb{E}_{\boldsymbol{x}}\left[\mathbb{V}_{\boldsymbol{x}'|\boldsymbol{x}}[\boldsymbol{x}']\right]$ is the covariance matrix $\mathbb{V}_{\boldsymbol{x}'|\boldsymbol{x}}[\boldsymbol{x}']$ of augmented views $\boldsymbol{x}'$ conditioned on $\boldsymbol{x}$, subsequently averaged over the data $\boldsymbol{x}$. Note that α_p and β reflect *multiplicative learning rate ratios* between the predictor and target networks relative to the online network. Finally, the terms involving η reflect weight decay.

As a gradient flow formulation, the learning rate α does not appear in Lemma 1. In the actual finite time update, the learning rate for W_p is $\alpha\alpha_p$, the EMA rate is $\alpha\beta = 1 - \gamma_{\rm a}$, where $\gamma_{\rm a}$ is the usual EMA parameter (e.g,. BYOL uses 0.996), and the weight decay for actual training is $\bar{\eta} := \alpha\eta$.

We note that since SimSiam is an ablation of BYOL that removes the EMA computation, the underlying dynamics of SimSiam can also be obtained from Lemma 1 simply by setting $W_a = W$, inserting this relation into Eqn. 2 and Eqn. 3, and ignoring Eqn. 4. Importantly, the stop-gradient on the target branch is still there.

Overall Eqns. 2-4 constitute our starting point for analyzing the combined roles of relative learning rates α_p and β, weight decay rate η and various ablations in determining the performance of both BYOL and SimSiam.

We first derive two very general results (see SM).

Theorem 1 (Weight decay promotes balancing of the predictor and online networks.)**.** *Completely independent of the particular dynamics of W_a in Eqn. 4, the update rules (Eqn. 2 and Eqn. 3) possess the invariance*

$$W(t)W^\intercal(t) = \alpha_p^{-1}W_p^\intercal(t)W_p(t) + e^{-2\eta t}C, \tag{5}$$

where C is a symmetric matrix that depends only on the initialization of W and W_p.

This theorem implies that for both BYOL and SimSiam, there exists a "balancing" that ensures that any matching between the online and target representations will not be attributable solely to the predictor weights, rendering the online weights useless. Instead what the predictor learns, the online network will also learn, which is important as the online network's representations are what is used for downstream tasks. We note that similar weight balancing dynamics has been discovered in multi-layer linear networks and matrix factorization (Arora et al., 2018; Du et al., 2018). Our results generalize this to SSL dynamics. Second, a nonzero weight decay could help remove the extra constant C due to initialization, further balancing the predictor and online network weights and possibly leading to better performance on downstream tasks (Tbl. 2).

EMA + no-bias	EMA + bias	no EMA + no-bias	no EMA + bias
70.62±1.05	70.99±1.01	71.36±0.44	71.37±0.77

Table 2. Top-1 accuracy of BYOL on STL-10 under linear evaluation protocol, trained for 100 epochs with no weight decay ($\eta = 0$) and $\alpha_p = 1$. It is worse than the baseline (74.51 ± 0.47 without predictor bias) when the weight decay is set to be $\eta = 0.0004$. "No-bias" means the linear predictor does not have a bias term.

Theorem 2 (The stop-gradient signal is essential for success.)**.** *With $W_{\rm a} = W$ (SimSiam case), removing the stop-gradient signal yields a gradient update for W given by positive semi-definite (PSD) matrix $H(t) := X' \otimes (W_p^\intercal W_p + I_{n_2}) + X \otimes \tilde{W}_p^\intercal\tilde{W}_p + \eta I_{n_1 n_2}$ (here $\tilde{W}_p := W_p - I_{n_2}$ and $\otimes$ is the Kronecker product):*

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathrm{vec}(W) = -H(t)\mathrm{vec}(W). \tag{6}$$

If the minimal eigenvalue $\lambda_{\min}(H(t))$ over time is bounded below, $\inf_{t\geq 0}\lambda_{\min}(H(t)) \geq \lambda_0 > 0$, then $W(t) \to 0$.

Thus we have proven analytically in this simple setting that removing the stop-gradient leads to representational collapse, as observed in more complex settings in SimSiam (Chen & He, 2020). Similarly, with $W_a = W$ and no predictor ($W_p = I_{n_2}$), then the dynamics Eqn. 3 also reduces to a similar form and $W(t) \to 0$ (see SM).

3. How multiple factors affect learning dynamics

The learning dynamics in Eqns. 2-4 constitute a set of high dimensional coupled nonlinear differential equations that

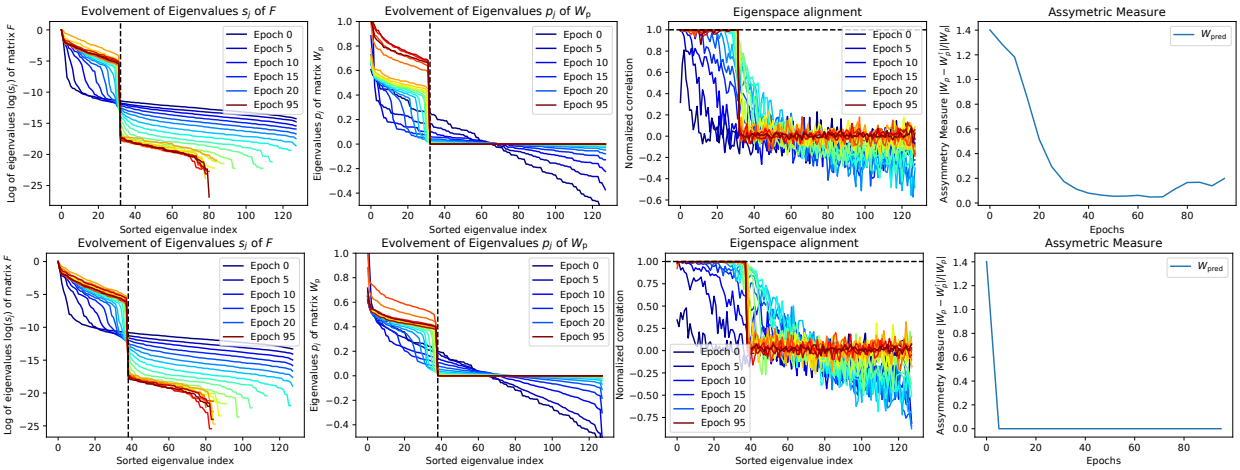

Figure 2. Training BYOL in STL-10 for 100 epochs with EMA. **Top row**: No symmetric regularization imposed on W_p, **Bottom row**: symmetric regularization on W_p. From left to right: **(1)** Evolvement of eigenvalues for F. Since F is PSD and its eigenvalue s_j varies across scales, we plot $\log(s_i)$. We could see some eigenvalues are growing while others are shrinking to zero over training. **(2)** Similar "step-function" behaviors for the predictor W_p. Its negative eigenvalues shrinks towards zero and leading eigenvalues becomes larger. **(3)** The eigenspace of F and W_p gradually align with each other (Theorem 3). For each eigenvector $\boldsymbol{u}_j$ of F, we compute cosine angle (normalized correlation) between $\boldsymbol{u}_j$ and $W_p\boldsymbol{u}_j$ to measure alignment. **(4)** W_p gradually becomes symmetric and PSD during training.

can be difficult to solve analytically in general. Therefore, to obtain analytic insights into the functional roles of the relative learning rates α_p and β and weight decay η, we make a series of simplifying assumptions. Intriguingly, under these simplifying assumptions we obtain a rich set of analytic predictions, which we then test experimentally in more realistic scenarios. We find, nicely, that these predictions still qualitatively hold *even when* our simplifying assumptions required for obtaining analytic results do not.

Assumption 1 (Proportional EMA)**.** *We first reduce the dimensionality of the dynamics in Eqns. 2-4 by enforcing that the target network W_a undergoes EMA but is forced to always be proportional to the online network via the relation $W_{\rm a}(t) = \tau(t)W(t)$. Inserting this relation into the EMA dynamics in Eqn. 4 yields $\dot{\tau}W + \tau\dot{W} = \beta(1-\tau)W$.*

Thus we obtain a reduced dynamics for W, W_p and τ. By not enforcing the stronger SimSiam constraint that $W_a = W$, we can still model EMA dynamics. Intuitively, $\tau = \tau(t)$ is a dynamic parameter that depends on how quickly $W = W(t)$ grows over time. If W is constant, then $\dot{W} = 0$ and τ stabilizes to 1. On the other hand, if W grows rapidly, then τ becomes small. While Assumption 1 is a simplification, as we shall see, it still reveals interesting verifiable predictions about the functional role of EMA.

Assumption 2 (Isotropic data and augmentation)**.** *We assume the data distribution $p(\boldsymbol{x})$ has zero mean and identity covariance, while the augmentation distribution $p_{\rm aug}(\cdot|\boldsymbol{x})$ has mean $\boldsymbol{x}$ and covariance $\sigma^2 I$. This simplifies the dynamics in Eqns. 2-4 by reducing the augmentation averaged data covariance to $X = I$ and the data averaged augmentation covariance to $X' = \sigma^2 I$.*

Many previous studies of deep learning dynamics made simplifying isotropic assumptions about data (Tian, 2017; Brutzkus & Globerson, 2017; Du et al., 2019; Bartlett et al., 2018; Safran & Shamir, 2018). Since our fundamental goal is to obtain the first analytic understanding of the dynamics of non-contrastive SSL methods, it is useful to first achieve this in the simplest possible isotropic setting. Interestingly, we will find that our final conclusions generalize to non-isotropic real world settings.

Assumption 3 (Symmetric predictor)**.** *We enforce symmetry in W_p by initializing it to be a symmetric matrix, and then symmetrizing the flow for W_p in Eqn. 2 (see SM).*

This symmetry assumption was motivated by both fixed point analysis and empirical findings. First, the fixed point of Eqn. 2 under Assumption 1 and 2 and $\eta > 0$ is always a symmetric matrix and in numerical simulation the asymmetric part $W_p - W_p^\intercal$ eventually vanishes (See Appendix for the proof and numerical simulations). Moreover, during BYOL training without a symmetry constraint on the predictor, W_p gradually moves towards symmetry (Fig. 2).

Second, a set of experiments reveal that whether the predictor is symmetric or not has a dramatic effect in terms of both performance and interaction with EMA. In our STL-10 experiment, enforcing symmetric W_p in the presence of EMA *improves* performance on downstream tasks (Tbl. 3). In contrast, in the absence of EMA, a symmetric W_p fails while an asymmetric W_p works reasonably well. Similar behavior holds on ImageNet: a symmetric one layer linear predictor W_p in SimSiam (i.e. without EMA) achieves performance no better than random guessing (Top-1/5: $0.1\%/0.5\%$), while an asymmetric 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

Table 3. The effect of symmetrization of W_p on downstream classification task (BYOL Top-1 on STL-10). Symmetric W_p leads to slightly better performance compared to regular W_p in the presence of EMA. On the other hand, without EMA, symmetric W_p crashes. Same effects happen in two-layer predictor with BatchNorm and ReLU as well. Weight decay $\bar{\eta} = 0.0004$ and $\alpha_p = 1$.

achieves a Top-1/5 accuracy of $68.1\%/88.2\%$. Our theory will explain this as well as show how to obtain good performance with a symmetric predictor without EMA by increasing its relative learning rate α_p.

3.1. Dynamical alignment of eigenspaces between the predictor and its input correlation matrix

Under the three assumptions stated above, we analyze the coupled dynamics of $F := WXW^\intercal$ and W_p. Note that F is the *correlation matrix* of the outputs of the online network which also serve as inputs to the predictor. By Assumption 2, $\mathbb{E}[\boldsymbol{x}] = \mathbf{0}$ and F is also the covariance matrix. We find F and W_p obey the following dynamics (see SM):

$$\begin{aligned} \dot{W}_p &= -\frac{\alpha_p}{2}(1+\sigma^2)\{W_p, F\} + \alpha_p \tau F - \eta W_p \quad (7) \\ \dot{F} &= -(1+\sigma^2)\{W_p^2, F\} + \tau\{W_p, F\} - 2\eta F \end{aligned}$$

This dynamics reveals that the eigenspace of W_p will gradually align with that of F under certain conditions (see SM for derivation):

Theorem 3 (Eigenspace alignment)**.** *Under Eqn. 7, the* commutator $[F, W_p] := FW_p - W_pF$ *satisfies:*

$$\frac{\mathrm{d}}{\mathrm{d}t}[F, W_p] = -[F, W_p]K - K[F, W_p] \quad (8)$$

where

$$K(t) = (1+\sigma^2)\left[\frac{\alpha_p}{2}F(t) + W_p^2(t) - \frac{\tau}{1+\sigma^2}W_p(t)\right] + \frac{3}{2}\eta I \quad (9)$$

If $\inf_{t\geq 0} \lambda_{\min}[K(t)] = \lambda_0 > 0$*, then the commutator*

$$\|[F(t), W_p(t)]\|_F \leq e^{-2\lambda_0 t}\|[F(0), W_p(0)]\|_F \to 0 \quad (10)$$

For symmetric W_p, when W_p and F commute they can be simultaneously diagonalized. Thus this shows that the eigenspace of W_p gradually aligns with that of F.

To test this prediction, we performed extensive experiments showing that training BYOL using ResNet-18 on STL-10 yields eigenspace alignment, as demonstrated in Fig. 2.

Now if the eigenspaces of W_p and F do align, we can obtain fully decoupled dynamics. Let the columns of the matrix U be the common eigenvectors, so that $W_p = U\Lambda_{W_p}U^\intercal$ where $\Lambda_{W_p} = \mathrm{diag}[p_1, p_2, \ldots, p_d]$, $F = U\Lambda_F U^\intercal$ where $\Lambda_F = \mathrm{diag}[s_1, s_2, \ldots, s_d]$. For each mode j, we have (see SM for derivation):

$$\begin{aligned} \dot{p}_j &= \alpha_p s_j\left[\tau - (1+\sigma^2)p_j\right] - \eta p_j \quad (11) \\ \dot{s}_j &= 2p_j s_j\left[\tau - (1+\sigma^2)p_j\right] - 2\eta s_j \quad (12) \\ s_j\dot{\tau} &= \beta(1-\tau)s_j - \tau\dot{s}_j/2. \quad (13) \end{aligned}$$

This decoupled dynamics constitutes a dramatically simplified set of 3 dimensional nonlinear dynamical systems for BYOL learning, and two dimensional nonlinear systems (obtained by constraining $\tau = 1$) for SimSiam. As expected, each mode's dynamics is equivalent to the 3 dimensional dynamics obtained by setting $n_1 = n_2 = 1$ in Eqns. 2-4 and making the replacements $W^2 = s_j$, $W_p = p_j$, and $W_a/W = \tau$ (see SM). Thus the decoupled dynamics in Eqns 11- 13 reduce to the scalar case of BYOL dynamics in Eqns. 2-4 after a change of variables and the condition in Thm. 3 reveals when this decoupled regime is reachable.

Non-symmetric W_p. When Assumption 3 is absent, the analysis is much more convoluted. One possible way is to decompose $W_p = A + B$ where $A = A^\intercal$ is symmetric and $B = -B^\intercal$ is skew-symmetric. We leave it for future work.

3.2. Analysis of decoupled dynamics

The simplified three (two) dimensional dynamics of BYOL (SimSiam) yields significant insights. First, there is clearly a collapsed fixed point at $p_j(t) = s_j(t) = 0$ and τ taking any value. We wish to understand conditions under which p_j and s_j can avoid this collapsed fixed point and grow from small random initial conditions. Since s_j is an eigenvalue of $WW^\intercal$, we are particularly interested in conditions under which s_j achieves large final values, corresponding to a non-collapsed online network, that are moreover sensitive to the statistics of the data, governed by σ^2.

Exact integral. First, an important observation, similar to Theorem 1, is that the dynamics possesses an exact integral of motion, obtained by multiplying Eqn. 11 by $2\alpha_p^{-1}p_j$, subtracting, Eqn. 12 and integrating over time yielding

$$s_j(t) = \alpha_p^{-1}p_j^2(t) + e^{-2\eta t}c_j \quad (14)$$

where $c_j = \alpha_p^{-1}p_j^2(0) - s_j(0)$ is fixed by initial conditions. In absence of weight decay ($\eta = 0$), this integral reveals that the initial condition encoded in c_j is never forgotten

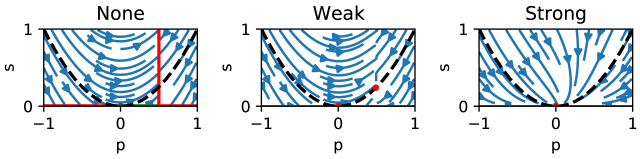

Figure 3. State space dynamics in Eqns. 11 and 12 for no ($\eta = 0$) weak ($\eta = 0.01$) and strong ($\eta = 1$) weight decay at fixed $\tau = 1$ and $\alpha_p = 1$. Red (green) points indicate stable (unstable) fixed points, blue curves indicate flow lines, and the dashed black curve indicates the parabola $s_j = p_j^2/\alpha_p$.

and the dynamics of p_j and s_j are confined to parabolas of the form $s_j(t) = p_j^2(t) + c_j$, as can be seen by the blue flow lines in Fig. 3(left). With weight decay ($\eta > 0$) over time the initial condition is forgotten and the dynamics approaches the invariant parabola $s_j = \alpha_p^{-1} p_j^2$ as can been seen by the approach of the blue flow lines to the black dashed parabola in Fig. 3 right and middle. We discuss these two cases in turn. First we note that in both cases, since the EMA computation is often very slow (Grill et al., 2020), corresponding to small β, the dynamics of τ in Eqn. 13 is slow relative to that of p_j and s_j. Therefore to understand the combined dynamics, we can search for the fixed points that p_j and s_j will rapidly approach at fixed τ. Over time τ will then either slowly approach 1 (BYOL) or be always equal to 1 (SimSiam), and s_j and p_j will follow their τ-dependent fixed points.

No weight decay. When $\eta = 0$, Eqns. 11 and 12 at a fixed value of τ yield a branch of collapsed fixed points given by $s_j = 0$ and p_j taking any value, and a branch of non-collapsed fixed points, with $p_j = \tau/(1+\sigma^2)$ and s_j taking any value (horizontal and vertical red/green lines in Fig. 3,left). A sufficient criterion on initial conditions to avoid the collapsed branch is $s_j(0) > p_j^2(0)/\alpha_p$ corresponding to lying above the dashed black parabola in Fig. 3,left. This restricted initial condition reveals why a fast predictor (large α_p) is advantageous (Obs#1): larger α_p leads to a smaller basin of attraction of the collapsed branch by flattening the dashed parabola. Indeed both BYOL and SimSiam have noted that a fast predictor can help avoid collapse. On the other hand, α_p cannot be infinitely large (Obs#2): since $s_j(+\infty) = s_j(0) + \alpha_p^{-1}(p_j^2(+\infty) - p_j^2(0))$, very large α_p implies that s_j, the final value of the online network characterizing the learned representation, does not grow even if p_j does. This is consistent with results which show that optimizing the predictor too often doesn't work in SimSiam (Chen & He, 2020), and directly setting an "optimal" predictor fails as well (Tbl. 1). The online network needs to grow along with the predictor and that cannot happen if the predictor is too fast.

Advantage of weight decay. In the non-collapsed branch of fixed points without weight decay (vertical red line in Fig. 3,left), the predictor p_j takes the exact value $\tau/(1+\sigma^2)$, which models the invariance to augmentation correctly: a large data augmentation variance σ^2 should lead to a small magnitude of the learned representation. Ideally, we want s_j to have the same property. With weight decay $\eta > 0$ in Eqn. 14, memory of the initial condition c_j fades away, yielding convergence to some point on the invariant parabola $s_j = \alpha_p^{-1} p_j^2$. (Obs#3): Therefore, by tying the online network to the predictor, weight decay allows s_j to also model invariance to augmentations correctly if the predictor does, regardless of the random initial condition c_j.

	Positive effects	Negative effects
Relative predictor lr α_p	#1,#6	#2
Weight decay η	#3,#7	#4,#5
EMA β	#8	#9,#10

Table 4. Summarization of positive/negative effects of various hyperparameter choices (EMA β, relative predictor learning rate α_p and weight decay η). "#1" means (Obs#1) in the text.

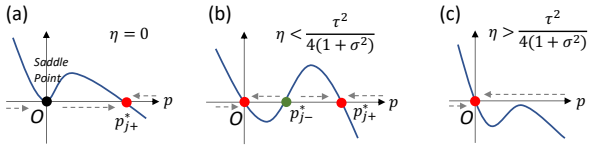

Figure 4. Fixed point of $\dot{p}_j = p_j(p_j - p_{j-}^*)(p_j - p_{j+}^*)$. Stable fixed points are in red, unstable in green and saddle in **black**. When the weight decay $\eta = 0$, the trivial solution $p_j = 0$ is a saddle. When $\eta > 0$, the trivial solution becomes stable near to the origin and initial p_j needs to be large enough to converge to the stable non-collapsed solution p_{j+}^*.

Dynamics on the invariant parabola. Because weight decay forces convergence to the invariant parabola $s_j = \alpha_p^{-1} p_j^2$, we next focus on dynamics along this parabola (i.e. $c_j = 0$ in Eqn. 14). In this case, Eqn. 13 has a solution:

$$\tau(t) = p_j^{-1}(t)\beta e^{-\beta t} \int_0^t p_j(t') e^{\beta t'} \mathrm{d}t, \qquad (15)$$

with initial condition $\tau(0) = 0$. Inserting the invariant $s_j = \alpha_p^{-1} p_j^2$ into Eqn. 11, the dynamics of p_j is given by:

$$\dot{p}_j = p_j^2 \left[\tau(t) - (1+\sigma^2) p_j\right] - \eta p_j. \qquad (16)$$

We first analyze the fixed points where $\dot{p}_j = 0$ at fixed τ. When the weight decay $0 < \eta \leq \frac{\tau^2}{4(1+\sigma^2)}$, p_j has has three fixed points (Fig. 4(b)):

$$p_{j\pm}^* = \frac{\tau \pm \sqrt{\tau^2 - 4\eta(1+\sigma^2)}}{2(1+\sigma^2)} > 0, \quad p_{j0}^* = 0$$

where both p_{j0}^* and p_{j+}^* are stable and p_{j-}^* is unstable, as shown in Fig. 4(b). The basin of attraction of the collapsed fixed point $p_{j0}^* = 0$ is $p_j < p_{j-}^*$ while the basin of attraction of the useful non-collapsed fixed point p_{j+}^* is $p_j > p_{j-}^*$, yielding an important constraint on initial conditions to avoid collapse. Note that p_{j-}^* is a decreasing function of τ and increasing function of η (see SM). This

means that with larger η, p^*_{j-} moves right and the basin of collapse expands (Obs#4). When $\eta > \frac{\tau^2}{4(1+\sigma^2)}$ there is only one stable fixed point $p^*_{j0} = 0$ (Fig. 4(c)). Under such strong weight decay collapse is unavoidable (Obs#5).

We now discuss the dynamics. First we define the quantity $\Delta_j := p_j[\tau - (1+\sigma^2)p_j] - \eta$, which must satisfy *two criteria*. Note that Eqn. 16 can be written as $\dot{p}_j = p_j \Delta_j$, so Δ_j must at some point be positive to drive $p_j(t)$ to any positive non-collapsed fixed point p^*_{j+}. Second, for eigenspace alignment in Theorem 3 to *remain* stable (even if the alignment has already happened), $K(t)$ must be positive definite (PD) in Eqn. 9. Using the eigen-space alignment conditions and the invariance $s_j = \alpha_p^{-1} p_j^2$, the positive definite condition on $K(t)$ can be written as

$$\Delta_j < \frac{1}{2}\left[\alpha_p(1+\sigma^2)s_j + \eta\right]. \tag{17}$$

This criterion and the criterion $\Delta_j > 0$ yield interesting insights into the roles of various hyperparameters choices.

First (Obs#6), larger predictor learning rate α_p can play an advantageous role by loosening the upper bound in Eqn. 17, making it easier to satisfy. Second (Obs#7), increasing η also has the same effect.

Role of EMA. Without EMA, $\tau \equiv 1$ and (Eqn. 17) may not hold initially when p_j is small. The reason is Δ_j is to leading order linear in p_j when $\tau = 1$ while the right hand side is to leading order $s_j \sim p_j^2$, so the left hand side has a larger contribution from p_j than the right.

EMA resolves this as follows. When the training begins, s_j is often quite small, and τ remains small since W changes rapidly. When p_j grows to the fixed point $p^*_{j+} \sim \tau/(1+\sigma^2)$, the growth of s_j stops, making τ *larger*. This in turns sets a higher fixed point goal for p_j. This process continues until the feature is stabilized and $\tau = 1$ (Fig. 5 for details).

Therefore, EMA can serve as an *automatic curriculum* (Obs#8): it sets an initial small goal of $\frac{\tau}{1+\sigma^2}$ for p_j so Δ_j need only be small and positive to both drive p_j larger and satisfy Eqn. 17. Then EMA gradually sets a higher goal for p_j by increasing τ, so that p_j and s_j can grow, while keeping the eigenspaces of W_p and F aligned.

As a trade-off, a very slow EMA schedule (β small) yields a slow training procedure (Obs#9) (See Fig. 5). Also small τ leads to larger p^*_{j-} and more eigen modes can be trapped in the collapsed basin (Obs#10).

3.3. Summarizing the effects of hyperparameters

We summarize the positive and negative effects of multiple hyperparameters in Tbl. 4. We next provide additional ablations and experiments to further justify our reasoning.

Different weight decay η_p and η_s. If we set a higher weight decay for the predictor (η_p) than the online net (η_s), then p_j grows slower than s_j and it is possible that the condition of Theorem 3 can still be satisfied without using EMA. Indeed Tbl. 5 shows this is the case.

	No predictor bias		With predictor bias	
	sym W_p	regular W_p	sym W_p	regular W_p
Weight decay only for predictor ($\bar{\eta}_p = 0.0004$ and $\bar{\eta}_s = 0$)				
EMA	71.91±0.70	70.54±0.93	73.67±0.47	70.89±0.98
no EMA	71.12±0.71	71.34±0.63	73.01±0.37	71.70±0.83
No weight decay for all ($\bar{\eta}_p = \bar{\eta}_s = 0$)				
EMA	71.76±0.28	70.62±1.05	71.86±0.39	70.99±1.01
no EMA	43.04±2.32	71.36±0.44	41.36±3.33	71.37±0.77

Table 5. Symmetric weight works without EMA, if we set weight decay for the predictor ($\bar{\eta}_p = 0.0004$) but not the trunk ($\bar{\eta}_s = 0$) in BYOL experiment on STL-10. Report Top-1 accuracy after 100 epochs. If there is no weight decay for *all layers*, then again symmetric weight doesn't work without EMA.

Larger learning rate of the predictor $\alpha_p > 1$. Our analysis predicts that one way to make symmetric W_p work with no EMA is to use $\alpha_p > 1$ (i.e. Theorem 3 is more easily satisfied). Fig. 6 verifies this prediction. Moreover Table 22 in Appendix of BYOL (Grill et al., 2020) also shows that $\alpha_p > 1$ is required to get BYOL working without EMA.

As a reference, Table 22 in Appendix I.2 of BYOL (Grill et al., 2020) also shows a similar trend: the learning rate of the (2-layer) predictor needs to be higher than that of the projector for strong performance in ImageNet, when EMA is absent.

4. Optimization-free Predictor W_p

A direct consequence of our theory is a new method for choosing the predictor that avoids gradient descent altogether. Instead, we estimate the correlation matrix F of predictor inputs and directly set W_p to be a function of this, thereby avoiding both the need to align the eigenspaces of F and W_p through optimization, and the need to initialize W_p outside the basin of collapse. As we shall see, this exceedingly simple, theory motivated method also yields better performance in practice compared to gradient-based optimization of a linear predictor.

We call our method **DirectPred** which simply estimates F, computes its eigen-decomposition $\hat{F} = \hat{U}\hat{\Lambda}_F\hat{U}^\intercal$, where $\hat{\Lambda}_F = \mathrm{diag}[s_1, s_2, \ldots, s_d]$, and sets W_p via

$$p_j = \sqrt{s_j} + \epsilon \max_j s_j, \quad W_p = \hat{U}\mathrm{diag}[p_j]\hat{U}^\intercal. \tag{18}$$

This choice is theoretically motivated by eigenspace-alignment between W_p and F (Theorem. 3) and convergence to the invariant parabola $s_j \propto p_j^2$ in Eqn. 14 with weight decay ($\eta > 0$). Here the estimate correlation matrix $\hat{F}$ can be obtained by a moving average:

$$\hat{F} = \rho\hat{F} + (1-\rho)\mathbb{E}_B\left[\boldsymbol{f}\boldsymbol{f}^\intercal\right] \tag{19}$$

where $\mathbb{E}_B[\cdot]$ is the expectation over a batch. Note that

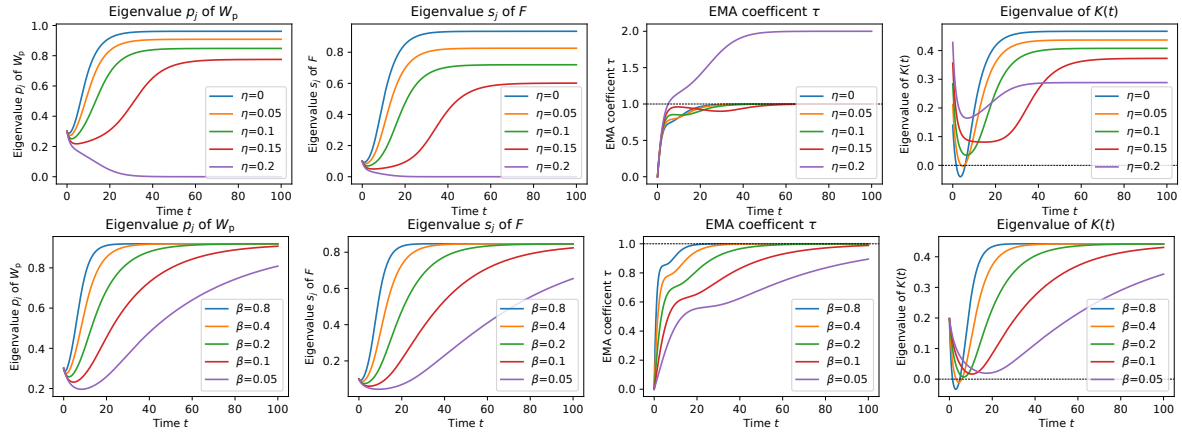

Figure 5. The role played by weight decay η and EMA β when applying symmetric regularization on W_p on synthetic experiments simulating decoupled dynamics (Eqn. 11-13). The learning rate $\alpha = 0.01$. Both terms boost the eigenvalue of $K(t)$ to above 0 so that eigen space alignment could happen (Theorem 3), but also come with different trade-offs. Here $\beta = 0.4$ so that $\alpha\beta = 0.004 = 1 - \gamma_a$ where $\gamma_a = 0.996$ as in BYOL. **Top row (Weight Decay η)**: A large η boost the eigenvalue of $K(t)$ up, but substantially decreases the final converging eigenvalues p_j and s_j (i.e., the final features are not salient), or even drags them to zero (no training happens). **Bottom row (EMA β)**. A small EMA β also boost the eigenvalue of $K(t)$, but the training converges much slower. Here $\eta = 0.04$ so that $\eta\alpha$ equals to the weight decay ($\bar{\eta} = 0.0004$) in our STL-10 experiments.

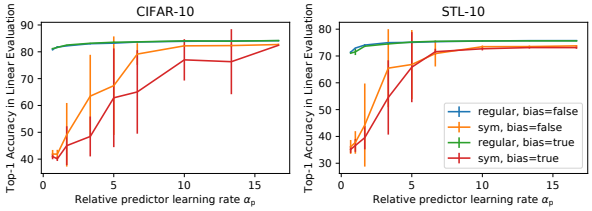

Figure 6. The effects of relative learning rate α_p without EMA. If $\alpha_p > 1$, symmetric W_p with no EMA can also work. Experiments on STL-10 and CIFAR-10 (Krizhevsky et al., 2009) (100 epochs with 5 random seeds).

	Regularization factor ϵ			
	0	0.01	0.1	0.5
$\rho = 0.3$	76.77±0.24	77.11±0.35	**77.86±0.16**	75.06±1.10
$\rho = 0.5$	76.65±0.20	76.76±0.33	**77.56±0.25**	75.22±0.81

Table 6. STL-10 Top-1 after BYOL training for 100 epochs, if we use **DirectPred** (Eqn. 18). It outperforms training W_p using gradient descent (74.51% in Tbl. 3, regular W_p with EMA). EMA is used in all experiments. No predictor bias. ρ defined in Eqn. 19.

	Initial constant c_j			
	0.1	0.05	−0.05	−0.1
freq=1	46.57±18.43	65.31±18.22	77.11±0.66	76.46±0.55
freq=2	75.01±0.48	75.10±0.35	76.83±0.52	76.31±0.27

Table 7. STL-10 Top-1 Accuracy after BYOL training for 100 epochs. With different c_j. $\rho = 0.3$ and $\epsilon = 0$. EMA is used in all experiments. No predictor bias.

where $\boldsymbol{f}$ is not zero-mean, we keep $\hat{F}$ a correlation matrix (rather than a covariance) *without* zero-centering $\boldsymbol{f}$, otherwise the performance deteriorates. We also added a regularization factor proportional to a small ϵ to boost the small eigenvalues s_j so they can learn faster. In all our experiments on real-world datasets, we use ℓ_2-normalization so the absolute magnitude of s_j doesn't matter.

Hyper-parameter `freq`. Besides, we also evaluate a hybrid approach by introducing `freq`, which is how frequently eigen-decomposition is conducted for matrix $\hat{F}$ to set W_p. For example, `freq = 5` means that eigen decomposition is run every 5 minibatches. When W_p is not set by eigen decomposition, it is updated by regular gradient updates. `freq = 1` means the eigen-decomposition is performed at every minibatch.

Tbl. 6 shows that directly computing W_p through **DirectPred** works *better* (76.77%) than training via gradient descent (74.51% in Tbl. 3, regular W_p with EMA). Additional regularization through ϵ yields even better performance (77.38%). Different ways to estimate F (moving average or simple average) yield only small differences.

The performance of **DirectPred** also remains good over many more training epochs (Tbl. 8). Moreover, if we allow some gradient steps in between directly setting W_p (i.e., `freq > 1`), performance becomes even better (80.28%). This might occur because the estimated $\hat{F}$ may not be accurate enough and SGD can help correct it. This also mitigates the computational cost of eigen-decomposition.

The constant c_j. What happens if $p_j = \sqrt{\max(s_j - c_j, 0)}$ with $c_j \neq 0$? If c_j is small negative, performance is still fine but a positive c_j leads to very poor performance (Tbl. 7), likely due to many small eigen-values s_j becoming zero and therefore trapped in the collapsed basin.

Feature-dependent W_p. Note one of the advantages of

	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

Table 8. STL-10/CIFAR-10 Top-1 accuracy of **DirectPred**, after training for longer epochs. $\rho = 0.3$, $\epsilon = 0.1$ with EMA.

BYOL variants	*Accuracy (60 ep)*		*Accuracy (300 ep)*	
	Top-1	Top-5	Top-1	Top-5
2-layer predictor*	$\mathbf{64.7}$	$\mathbf{85.8}$	$\mathbf{72.5}$	90.8
linear predictor	59.4	82.3	69.9	89.6
DirectPred	64.4	$\mathbf{85.8}$	72.4	$\mathbf{91.0}$

* 2-layer predictor is BYOL default setting.

Table 9. ImageNet experiments comparing **DirectPred** with BYOL (Grill et al., 2020). *Without* gradient-based training, **DirectPred** is able to match the performance of the default 2-layer predictor introduced by BYOL, and significantly outperform the linear predictor by 5% (60 epoch) and 2.5% (300 epoch).

using two layer predictors is that W_p can depend on the input features. We explored this idea by using a few random partitions of the input space, and within each random partition we estimated a different correlation matrix $\hat{F}$. The final $\hat{F}$ is the sum of all the correlation matrices. With 6 random partitions, **DirectPred** achieves 78.20 ± 0.16 Top-1 accuracy after 100 epochs, closing performance gap to two-layer predictors (78.85% in Tbl. 3). We leave a thorough analysis of the two layer setting to future work.

ImageNet experiments. We conducted additional experiments on ImageNet (Deng et al., 2009), with our own BYOL (Grill et al., 2020) implementation. We used ResNet-50 (He et al., 2016) as the backbone to produce features for a linear probe, followed by a projector and a predictor. The architecture design (e.g., feature dimensions), augmentation strategies (e.g., color jittering, blur (Chen et al., 2020a), solarization, etc.) and linear classification protocol strictly follow BYOL (Grill et al., 2020).

We experimented with two different training settings to study the generalization ability of **DirectPred**. In the first setting, we employ an asymmetric loss (given two views, only one view is used as the prediction target). The loss is optimized using standard SGD for 60 epochs with a batch size of 256. The second setting follows BYOL more closely, where we use a symmetrized loss, 4096 batch size and LARS optimizer (You et al., 2017), and train for 300 epochs.

The results are summarized in Tbl. 9. Both settings exhibit similar behaviors in comparison, and we take the 300-epoch results as our highlights in the following. As a baseline, the default 2-layer predictor from BYOL (with BatchNorm and ReLU, 4096 hidden dimension, 256 input/output dimension) achieves 72.5% top-1 accuracy, and 90.8% top-5 accuracy with 300-epoch pre-training. This reproduces the accuracy reported in BYOL (Grill et al., 2020). We find **DirectPred** can match this performance (72.4% top-1, and 91.0% top-5) *without* any gradient-based training by instead directly setting the (256×256) linear predictor weights every mini-batch. In particular for top-5 **DirectPred** is even 0.2% better. For a fair comparison, we also run BYOL with a learned linear predictor. We find the performance drops to 69.9%, and 89.6% respectively (2.5% gap to our method). The gap is even bigger in 60-epoch settings, up to 5.0% in top-1 (59.4% vs. 64.4%). These experiments demonstrate the success of **DirectPred** on STL-10 and CIFAR can also generalize and scale to ImageNet.

5. Discussion

Summary. Therefore, remarkably, our theoretical analysis of non-contrastive SSL, primarily centered around a 3 dimensional nonlinear dynamical system, not only yields conceptual insights into the functional roles of complex ingredients like EMA, stop-gradients, predictors, predictor symmetry, diverse learning rates, weight decay and all their interactions, but also predicts the performance patterns of many ablation studies as well as suggests an exceedingly simple **DirectPred** method that rivals the performance of more complex predictor dynamics in real-world settings.

Two-layer non-linear predictor. With only a linear predictor, our results on ImageNet (Tbl. 9) have already shown strong performance, on par with a default BYOL setting with a 2-layer predictor on ImageNet. One interesting question is how the dynamics changes if the predictor has 2 layers. While we don't provide a formal analysis and the math can be quite complicated, the intuition here is that the "fat" 2-layer predictor used in practice (e.g., more (4096) hidden dimension than input/output dimensions (256), and a ReLU in between) essentially provides a large pool of initial weight directions to start with, and some of them could be "lucky draws", that make eigen-space alignment faster. On the other hand, a 1-layer predictor with gradient updates may get stuck in local minima. Therefore, with the same number of epochs, a 2-layer predictor outperforms 1-layer, and is comparable with **DirectPred** which does not suffer from local minima issues.

Acknowledgements

We thank Lantao Yu for helpful discussions.

References

Arora, S., Cohen, N., and Hazan, E. On the optimization of deep networks: Implicit acceleration by overparameterization. In *ICML*. PMLR, 2018.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. A theoretical analysis of contrastive unsupervised representation learning. 2019.

Bachman, P., Hjelm, R. D., and Buchwalter, W. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

Bartlett, P., Helmbold, D., and Long, P. Gradient descent with identity initialization efficiently learns positive definite linear transformations by deep residual networks. In *ICML*, 2018.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. Signature verification using a" siamese" time delay neural network. *NeurIPS*, 1994.

Brutzkus, A. and Globerson, A. Globally optimal gradient descent for a convnet with gaussian inputs. In *ICML*, 2017.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020a.

Chen, X. and He, K. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020.

Chen, X., Fan, H., Girshick, R., and He, K. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020b.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics*, 2011.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Du, S. and Hu, W. Width provably matters in optimization for deep linear neural networks. In *ICML*, 2019.

Du, S. S., Hu, W., and Lee, J. D. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *arXiv preprint arXiv:1806.00900*, 2018.

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. *ICML*, 2019.

Fetterman, A. and Albrecht, J. Understanding self-supervised and contrastive learning with "bootstrap your own latent" (byol), 2020. `https://untitled-ai.github.io/understanding-self-supervised-contrastive-learning.html`.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Kawaguchi, K. Deep learning without poor local minima. *NeurIPS*, 2016.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Lampinen, A. K. and Ganguli, S. An analytic theory of generalization dynamics and transfer learning in deep linear networks. In *ICLR*, 2018.

Laurent, T. and Brecht, J. Deep linear networks with arbitrary loss: All local minima are global. In *ICML*, pp. 2902–2907. PMLR, 2018.

Lee, J. D., Lei, Q., Saunshi, N., and Zhuo, J. Predicting what you already know helps: Provable self-supervised learning. *arXiv preprint arXiv:2008.01064*, 2020.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Pennington, J., Schoenholz, S., and Ganguli, S. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In *NeurIPS*. 2017.

Pennington, J., Schoenholz, S. S., and Ganguli, S. The emergence of spectral universality in deep networks. In *AISTATS*, 2018.

Safran, I. and Shamir, O. Spurious local minima are common in two-layer relu neural networks. In *ICML*. PMLR, 2018.

Saxe, A. M., McClelland, J. L., and Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Saxe, A. M., McClelland, J. L., and Ganguli, S. A mathematical theory of semantic development in deep neural networks. *Proc. Natl. Acad. Sci. U. S. A.*, 2019.

Tian, Y. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. In *ICML*, 2017.

Tian, Y., Krishnan, D., and Isola, P. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

Tosh, C., Krishnamurthy, A., and Hsu, D. Contrastive learning, multi-view redundancy, and linear models. *arXiv preprint arXiv:2008.10150*, 2020.

You, Y., Gitman, I., and Ginsburg, B. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.