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/luckmatters/tree/master/ssl](https://github.com/facebookresearch/luckmatters/tree/master/ssl)

ICML 2021 Long oral
Self-supervised Learning (SimCLR)

Self-supervised Learning (SimCLR)

Why a good representation can be learned this way?
Roles played by Neural Network, Data Augmentation and Training Procedure?

Data Augmentation
\[ x_1, x_2 \sim p_{\text{aug}}(\cdot | x) \]

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)?

Why do they not collapse to trivial solutions?

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:

\[ J(W, W_p) := \frac{1}{2} \mathbb{E}_{x_1, x_2} \left[ \| W_p f_1 - \text{StopGrad}(f_{2a}) \|_2^2 \right] \]

Linear online network \( W \)

Linear target network \( W_a \)

Linear predictor \( W_p \)
Learning Dynamics

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

\[
\begin{align*}
\dot{W}_p &= \alpha_p \left(-W_p W (X + X') + W_a X\right) W^\top - \eta W_p \\
\dot{W} &= W_p^\top \left(-W_p W (X + X') + W_a X\right) - \eta W \\
\dot{W}_a &= \beta (-W_a + W)
\end{align*}
\]

<table>
<thead>
<tr>
<th>Hyper-parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\alpha_p)</td>
<td>Relative learning rate of the predictor</td>
</tr>
<tr>
<td>(\eta)</td>
<td>Weight decay</td>
</tr>
<tr>
<td>(\beta)</td>
<td>The rate of Exponential Moving Average (EMA)</td>
</tr>
</tbody>
</table>
Theorem 2: No Stop-Gradient doesn’t work ($\mathcal{W} \to 0$)

\[
\frac{d}{dt} \text{vec}(\mathcal{W}) = - \left[ X' \otimes (\mathcal{W}_p^T \mathcal{W}_p + I) + X \otimes \mathcal{\bar{W}}_p^T \mathcal{\bar{W}}_p \right] \text{vec}(\mathcal{W})
\]

Here $\mathcal{\bar{W}}_p := \mathcal{W}_p - I$
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 $W_p$

<table>
<thead>
<tr>
<th>No predictor bias</th>
<th>With predictor bias</th>
</tr>
</thead>
<tbody>
<tr>
<td>sym $W_p$</td>
<td>regular $W_p$</td>
</tr>
<tr>
<td>sym $W_p$</td>
<td>regular $W_p$</td>
</tr>
</tbody>
</table>

**One-layer linear predictor**

<table>
<thead>
<tr>
<th>EMA</th>
<th>No EMA</th>
<th>EMA</th>
<th>No EMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>75.09 ± 0.48</td>
<td>74.51 ± 0.47</td>
<td>74.16 ± 0.33</td>
<td>74.51 ± 0.47</td>
</tr>
<tr>
<td><strong>36.62 ± 1.85</strong></td>
<td><strong>72.85 ± 0.16</strong></td>
<td><strong>36.04 ± 2.74</strong></td>
<td><strong>72.13 ± 0.53</strong></td>
</tr>
</tbody>
</table>

**Two-layer predictor with BatchNorm and ReLU**

<table>
<thead>
<tr>
<th>EMA</th>
<th>No EMA</th>
<th>EMA</th>
<th>No EMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>71.58 ± 6.46</td>
<td>78.85 ± 0.25</td>
<td>78.53 ± 0.34</td>
<td>77.64 ± 0.41</td>
</tr>
<tr>
<td><strong>35.59 ± 2.10</strong></td>
<td><strong>65.98 ± 0.71</strong></td>
<td><strong>41.92 ± 4.25</strong></td>
<td><strong>65.59 ± 0.66</strong></td>
</tr>
</tbody>
</table>

Symmetric $W_p$ affects the performance a lot!
Symmetrized Dynamics

Define anti-commutator \( \{A, B\} := AB + BA \):

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

Here \( F := E[ff^T] = WXW^T \) is the correlation matrix of the input of the predictor.
Eigenspace Alignment

**Theorem 3**: Under certain conditions,

\[
[F, W_p] := FW_p - W_p F \to 0 \text{ when } t \to +\infty
\]

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:

\[ \dot{p}_j = \alpha p_j s_j \left[ \tau - (1 + \sigma^2)p_j \right] - \eta p_j \]
\[ \dot{s}_j = 2p_j s_j \left[ \tau - (1 + \sigma^2)p_j \right] - 2\eta s_j \]
\[ s_j \dot{\tau} = \beta (1 - \tau) s_j - \tau \dot{s}_j / 2. \]

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?

(a) $\eta = 0$
- No Weight Decay

(b) $\eta < \frac{\tau^2}{4(1 + \sigma^2)}$
- Weak Weight Decay
- Stable Trivial
- Stable Nontrivial

(c) $\eta > \frac{\tau^2}{4(1 + \sigma^2)}$
- Strong Weight Decay

Saddle Point

$p_j^+$

$p_j^-$

$p_0$

O
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 $\alpha_p$

**Positive 😊**
1. Large $\alpha_p$ shrinks the size of trivial basin
2. Relax the condition of eigenspace alignment

**Negative 😞** With very large $\alpha_p$, eigenvalue of $F$ won’t grow (and no feature learning)
Exponential Moving Average rate $\beta$

$\beta$ large $\rightarrow W_a(t)$ catches $W(t)$ faster

**Positive 😊:** Slower rate (small $\beta$) relaxes the condition of eigenspace alignment

$\tau$ 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*  *second order*  *second order*

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

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

Guaranteed Eigenspace Alignment 😊
# Performance of DirectPred on STL-10/CIFAR-10

<table>
<thead>
<tr>
<th>Downstream Classification Top-1</th>
<th>Number of epochs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>STL-10</td>
</tr>
<tr>
<td>DirectPred</td>
<td>77.86 ± 0.16</td>
</tr>
<tr>
<td>DirectPred (freq=5)</td>
<td>77.54 ± 0.11</td>
</tr>
<tr>
<td>SGD baseline</td>
<td>75.06 ± 0.52</td>
</tr>
<tr>
<td></td>
<td>85.21 ± 0.23</td>
</tr>
<tr>
<td>DirectPred (freq=5)</td>
<td>84.93 ± 0.29</td>
</tr>
<tr>
<td>SGD baseline</td>
<td>84.49 ± 0.20</td>
</tr>
</tbody>
</table>

---

Facebook Artificial Intelligence
## Performance of DirectPred on ImageNet

### ImageNet performance (60 epoch)

<table>
<thead>
<tr>
<th>BYOL variants</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Top-1</td>
</tr>
<tr>
<td>2-layer predictor (default)</td>
<td>64.7</td>
</tr>
<tr>
<td>linear predictor</td>
<td>59.4</td>
</tr>
<tr>
<td>DirectPred</td>
<td>64.4</td>
</tr>
</tbody>
</table>

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)

<table>
<thead>
<tr>
<th>BYOL variants</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Top-1</td>
</tr>
<tr>
<td>2-layer predictor (default)</td>
<td>72.5</td>
</tr>
<tr>
<td>linear predictor</td>
<td>69.9</td>
</tr>
<tr>
<td>DirectPred</td>
<td>72.4</td>
</tr>
</tbody>
</table>

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