TOWARDS A UNIFIED THEORETICAL UNDERSTANDING OF NON-CONTRASTIVE LEARNING VIA RANK DIFFERENTIAL MECHANISM

Anonymous authors
Paper under double-blind review

ABSTRACT

Recently, a lot of advances in self-supervised visual learning are brought about by contrastive learning that aligns positive pairs while pushing negative pairs apart. Surprisingly, a variety of new methods, such as BYOL, SimSiam, SwAV, DINO, shows that when equipped with some architectural asymmetric designs, aligning positive pairs alone is sufficient to attain good performance. However, it is still not fully clear how these seemingly different asymmetric designs can avoid feature collapse. Despite some understandings of some specific modules (like the predictor in BYOL), there is yet no unified theoretical understanding, particularly for those who also work without the predictor (like DINO). In this work, we propose a new understanding for non-contrastive learning, named the Rank Differential Mechanism (RDM). We show that these asymmetric designs all create a consistent difference in the dual-branch outputs as measured by their effective rank. This rank difference will provably lead to an improvement of effective dimensionality and alleviate either complete or dimensional feature collapse. Different from previous theories, our RDM theory is applicable to different asymmetric designs (with and without the predictor), and thus can serve as a unified understanding of existing non-contrastive learning methods. Besides, our RDM theory also provides practical guidelines for designing many new non-contrastive variants. We show that these variants indeed achieve comparable performance to existing methods on benchmark datasets, and some of them even outperform the baselines.

1 INTRODUCTION

Self-supervised learning of visual representations undergoes rapid progress in recent years, particularly due to the rise of contrastive learning (CL) (Oord et al., 2018). Canonical contrastive learning methods like SimCLR (Chen et al., 2020) and MoCo (He et al., 2020) utilize both positive samples (for feature alignment) and negative samples (for feature uniformity). Surprisingly, researchers notice that CL can also work well by only aligning positive samples, which is often referred to as non-contrastive learning. Without the help of negative samples, various techniques are proposed to prevent feature collapse, for example, stop-gradient, momentum encoder, predictor (BYOL (Grill et al., 2020), SimSiam (Chen & He, 2021)), Sinkhorn iterations (SwAV (Caron et al., 2020)), feature centering and sharpening (DINO (Caron et al., 2021)). These above designs all create a certain of *asymmetry* between the online branch (with gradient) and the target branch (without gradient) (Wang et al., 2022a). Empirically, these tricks can successfully alleviate feature collapse and obtain comparable or even superior performance than canonical contrastive learning. Despite this progress, it is still not clear why these different heuristics can reach the same goal.

Some existing works are proposed to understand some specific non-contrastive techniques, mostly focusing on the predictor head proposed by BYOL (Grill et al., 2020). From an empirical side, Chen & He (2021) think that the predictor helps approximate the expectation over augmentations, and Zhang et al. (2022) take a center-residual decomposition of representations for analyzing the collapse. From a theoretical perspective, Tian et al. (2021) analyze the dynamics of predictor weights under simple linear networks, and Wen & Li (2022) obtain optimization guarantees for two-layer nonlinear networks. These theoretical discussions often need strong assumptions on the data distribution (*e.g.,* standard normal (Tian et al., 2021)) and augmentations (*e.g.,* random masking (Wen & Li, 2022)). Besides, their analyses are often problem specific, which is hardly extendable to other non-contrastive variants without a predictor, *e.g.,* DINO. Therefore, a natural question is raised here:

Are there any basic principles behind these seemingly different techniques?

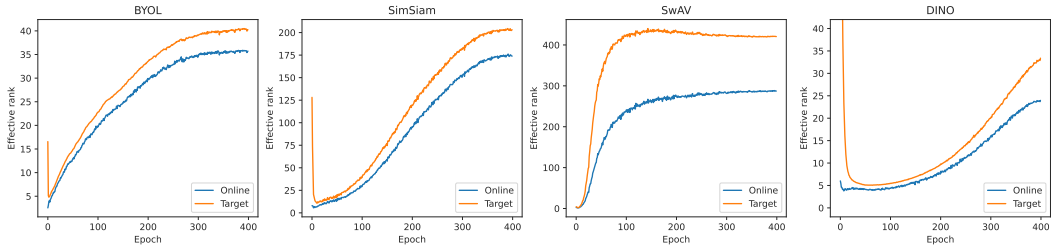


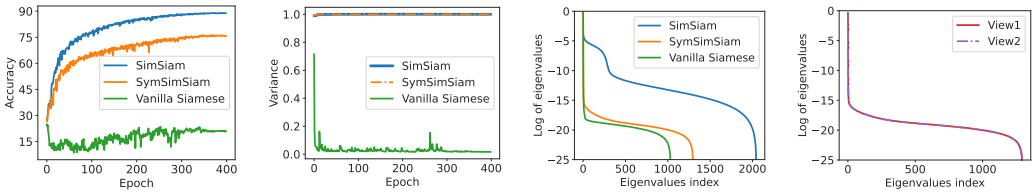
Figure 1: The effective rank of the normalized outputs of the online and target branches for four different non-contrastive methods (BYOL, SimSiam, SwAV, and DINO) on CIFAR-10.

In this paper, we make the first attempt in this direction by discovering a common mechanism behind these non-contrastive variants. To get a glimpse of it, in Figure 1, we measure the effective rank (Roy & Vetterli, 2007) of four different non-contrastive methods (BYOL, SimSiam, SwAV, and DINO). We find the following phenomenons: 1) among different methods, the target branch (orange line) has consistently higher rank than the online branch (blue line); and 2) after the initial warmup stage, the rank of the online branch (blue line) consistently improves along the training process. Inspired by this observation, we propose a new theoretical understanding of non-contrastive methods, dubbed Rank Differential Mechanism (RDM), where we show that these different techniques essentially create a *positive rank difference* between the target and the online branches. Through this rank difference, the alignment with the target output will help improve the output rank of the online encoder and thus alleviate feature collapse.[1] As a result, non-contrastive methods can implicitly improve the feature diversity without explicit uniformity regularization and attain similar performance as canonical CL methods.

In the rest of this paper, we are devoted to providing theoretical and empirical justifications for the proposed RDM theory from the following aspects:

- **The role of asymmetry: symmetry can alleviate complete collapse while asymmetry can further alleviate dimensional collapse.** Contrary to the common belief that asymmetry is necessary for avoiding complete collapse, we show that a symmetric architecture can provably alleviate complete collapse, while it still suffers from dimensional collapse. This reminds us to explore how asymmetric designs alleviate dimensional collapse.
- **The proposed RDM could explain how asymmetric designs avoid dimensional collapse via the ladder of rank difference.** Through a deeper investigation of rank difference, we notice that it is generated by the low-pass online filtering (or equivalently, a high-pass target filtering) effect of the asymmetric designs. Based on this observation, we theoretically prove that the low-pass filter can produce the rank difference. The existence of rank difference will further improve the effective feature dimensionality with the alignment loss, and alleviate dimensional collapse along training.
- **How existing asymmetric designs produce RDM: understanding online learnable predictors (BYOL, SimSiam) and target feature transformations (SwAV, DINO).** Given the efficacy of RDM, we further explore how existing asymmetric designs produce the rank difference. For SimSiam and BYOL, we theoretically show that the low-pass filtering effect of the online predictor is induced by high-frequency noise generated by data augmentations. For in SwAV and DINO, we also show that each of their transformations generate a higher rank on the target output, and lead to a high-pass filter effect on the target branch.
- **Application of RDM: inspiring new asymmetric designs that also work well in practice.** Following the principle of RDM, we design a series of non-contrastive variants to empirically verify the effectiveness of our theory. For the online encoder, we show that different variants of low-pass filters can also attain fairly good performance. We also design a new kind of *target predictors with high-pass filters*. Experiments show that SimSiam with our target predictors can outperform DirectPred (Tian et al., 2021) and achieve comparable or even superior performance to the original SimSiam.

[1] There are two kinds of feature collapse: one is the complete collapse where the embedding vectors collapse to the same point, the other one is the milder dimensional collapse where the embedding vectors only span a lower dimensional space. Features without dimensional collapse certainly do not have complete collapse.



(a) Accuracy (b) Variance (c) Eigenvalues (d) Two branches

Figure 2: Comparison of the asymmetric SimSiam with two symmetric baselines, SymSimSiam (ours) and Vanilla Siamese on CIFAR-10, where SymSimSiam adopts $\mathcal{L}_{\text{sym}}(f'_\theta)$ (Eq. 2), while Vanilla Siamese adopts $\mathcal{L}_{\text{sym}}(f_\theta)$. (a): Linear probing test accuracy. (b): Feature variance of normalized outputs. (c): Eigenvalues of the correlation matrix of the normalized outputs. (d): Eigenvalues of the correlation matrix of the normalized outputs from two branches of SymSimSiam.

2 Related Work

Non-contrastive Learning. Among existing methods, BYOL (Grill et al., 2020) is the first to show we can alleviate the feature collapse of aligning positive samples along with an online predictor and a momentum encoder. Later, SimSiam (Chen & He, 2021) further simplifies this requirement and shows that only the online predictor is enough. As for another thread, SwAV (Caron et al., 2020) applies Sinkhorn-Knopp iterations (Cuturi, 2013) to the target output from an optimal transport view. DINO (Caron et al., 2021) further simplifies this approach by simply combining feature centering and feature sharpening. Remarkably, all these methods adopt an online-target dual-branch architecture and gradients from the target branch are detached. Our theory provides a unified understanding of these designs and reveals their common underlying mechanisms.

Dimensional Collapse of Self-supervised Representations. Prior to ours, several works also explore the dimensional collapse issue in contrastive learning. Ermolov et al. (2021) and Hua et al. (2021) propose feature whitening techniques to alleviate dimensional collapse, similar in spirit to Barlow Twins (Zbontar et al., 2021) with a feature decorrelation regularization. Jing et al. (2021) point out the dimensional collapse of contrastive learning without using the projector, and propose DirectCLR as a direct replacement. Instead, our work mainly focuses on understanding how asymmetric designs help alleviate dimensional collapse in non-contrastive learning.

Theoretical Analysis on Contrastive Learning. Saunshi et al. (2019) first establish downstream guarantees for contrastive learning, which are later gradually refined (Nozawa & Sato, 2021; Ash et al., 2022; Bao et al., 2022). Tosh et al. (2021) and Lee et al. (2021) also propose similar guarantees on downstream tasks. However, these methods mostly rely on the conditional independence assumption that is far from practice. Recently, HaoChen et al. (2021) and Wang et al. (2022b) propose an augmentation graph perspective with more practical assumptions, and contribute the generalization ability to the existence of augmentation overlap (which also exists for non-contrastive learning). Wen & Li (2021) analyze the feature dynamics of contrastive learning with shadow ReLU networks.

3 Asymmetry is the Key to Alleviate Dimensional Collapse

Prior works tend to believe that asymmetric designs are necessary for avoiding complete feature collapse (Zhang et al., 2022), while we show that a *fully symmetric* architecture, dubbed SymSimSiam (Symmetric Simple Siamese network), can also avoid complete collapse. Specifically, we simply align the positive pair (x, x^+) with a symmetric alignment loss,

$$\mathcal{L}_{\text{sym}}(f'_\theta) = -\mathbb{E}_{x,x^+} f'_\theta(x)^\top f'_\theta(x^+), \tag{1}$$

where we apply feature centering on the output of an encoder f, *i.e.,* $f'_\theta(\cdot) = f_\theta(\cdot) - \mu$. The feature average $\mu = \mathbb{E}_x f_\theta(x)$ can be computed via mini-batch estimate or exponential moving average as in Batch Normalization (Ioffe & Szegedy, 2015) (see Algorithm 1). The following theorem states that SymSimSiam can avoid complete feature collapse by simultaneously maximizing total feature variance $\text{Var}(f_\theta(x))$.

Theorem 1. *When $f_\theta(x)$ is ℓ_2-normalized, the SymSimSiam objective is equivalent to*

$$\mathcal{L}_{\text{sym}}(f'_\theta) = \mathcal{L}_{\text{sym}}(f_\theta) - \text{Var}(f_\theta(x)) + 1 = -\mathbb{E}_{x,x^+} f_\theta(x)^\top f_\theta(x^+) - \mathbb{E}_x \|f_\theta(x) - \mu\|^2 + 1. \tag{2}$$

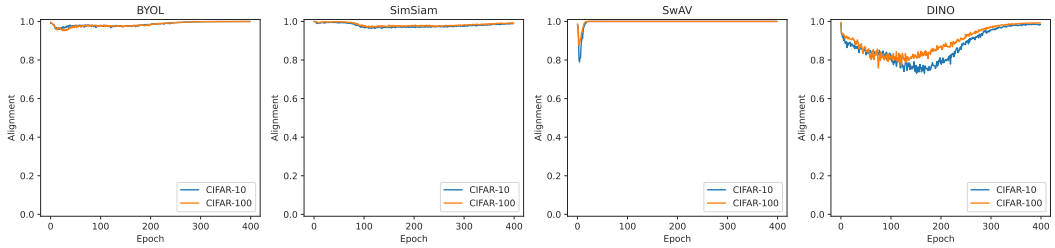


Figure 3: Eigenspace alignment of the online and target outputs along training. Given u_i as an eigenvector of $\mathbb{C}_z$, if it is also an eigenvector of $\mathbb{C}_p$, then $\mathbb{C}_p u_i = \lambda_i' u_i$ is in the same direction as u_i. Thus, we measure the alignment of u_i by computing the cosine similarity between u_i and $\mathbb{C}_p u_i$, and take the average over each u_i as the overall eigenspace alignment (details in Appendix B.3).

Empirically shown in Figures 2(a) & 2(b), compared to vanilla Siamese network, SymSimSiam indeed alleviates complete collapse, *i.e.,* the feature variance is maximized along training and a good linear probing accuracy is achieved. However, the accuracy of SymSimSiam is also obviously lower than the asymmetric SimSiam (Figure 2(a)). This indicates that symmetric design could alleviate complete collapse, but it may be *not* enough to prevent *dimensional feature collapse*. Intuitively, features uniformly distributed on a great circle of a unit sphere have maximal variance while being dimensionally collapsed. As further shown in Figure 2(c), SymSimSiam indeed suffers from more severe dimensional collapse compared to SimSiam. With few effective dimensions, the encoder network has limited ability to encode rich semantics (HaoChen et al., 2021).

The above SymSimSiam experiments show that with a symmetric architecture, we can easily prevent complete collapse while hardly improving the effective feature dimensionality for overcoming dimensional collapse. Instead, we also notice that SimSiam with asymmetric designs can alleviate dimensional collapse and achieve better performance. It reflects a fact that the asymmetry in existing non-contrastive methods is the key to alleviating *dimensional collapse*, which leads us to the main focus of our paper, *i.e.,* demystifying asymmetric designs.

4 Asymmetry Alleviates Dimensional Collapse via Rank Difference

In Figure 1, we have observed a common mechanism behind the non-contrastive learning methods: these asymmetric designs create *positive rank differences* between target and online outputs consistently throughout training. Here, we provide a formal analysis of this phenomenon from both theoretical and empirical sides and show how it helps alleviate dimensional feature collapse.

4.1 Problem Setup

Given a set of natural training data $\bar{\mathcal{X}} = \{\bar{x} \mid \bar{x} \in \mathbb{R}^d\}$, we can draw a pair of positive samples (x, x^+) from independent random augmentations of a natural example $\bar{x}$ with distribution $\mathcal{A}(\cdot|\bar{x})$. Their joint distribution satisfies $\mathcal{P}(x, x^+) = \mathcal{P}(x^+, x) = \mathbb{E}_{\bar{x}} \mathcal{A}(x|\bar{x})\mathcal{A}(x^+|\bar{x})$. Without loss of generality, we consider a finite sample space $|\mathcal{X}| = n$ (can be exponentially large) following HaoChen et al. (2021), and denote the collection of online and target outputs as $p, z \in \mathbb{R}^{n \times k}$ whose x-th row is p_x, z_x, respectively. We consider a general alignment loss $L(p) = \mathbb{E}_{x,x^+} \ell(p_x, \mathrm{sg}(z_{x^+}))$ to cover different variants of non-contrastive methods. The online and target outputs p_x, z_x are either ℓ_2-normalized (BYOL and SimSiam) or $\mathrm{softmax}$-normalized (SwAV and DINO). For the loss function ℓ, BYOL and SimSiam adopt the mean square error (MSE) loss, while SwAV and DINO adopt the cross entropy (CE) loss. The operator $\mathrm{sg}(\cdot)$ denotes stopping the gradients from the operand.

In this work, we measure the effective feature dimensionality by the effective rank (erank) (Roy & Vetterli, 2007) of the population feature correlation matrix $\mathbb{C} = \mathbb{E}_x f(x) f(x)^\top$. Denote its eigendecomposition as $\mathbb{C} = V \Lambda V^\top$, where V contains eigenvectors in columns, and Λ is the diagonal matrix with descending eigenvalues $\lambda_1 \geq \cdots \geq \lambda_k \geq 0$. The effective rank of $\mathbb{C}$ is defined as $\mathrm{erank}(\mathbb{C}) = \exp(H(q))$, where $q = (q_1, \ldots, q_k), q_i = \lambda_i / \sum_i \lambda_i$ are the normalized eigenvalues as a probability distribution, and $H(q) = -\sum_i q_i \log(q_i)$ is its Shannon entropy. Compared to the canonical rank, the effective rank is real-valued and invariant to feature scaling. A more uniform distribution of eigenvalues has a larger effective rank, and vice versa. Thus, the effective rank of $\mathbb{C}$ is a proper metric to measure the degree of dimensional feature collapse.

4.2 THE RANK DIFFERENTIAL MECHANISM

Here, we provide a new understanding of non-contrastive learning, Rank Differential Mechanism, through the following three mechanisms shared by different non-contrastive methods.

Mechanism 1 (Eigenspace Alignment)**.** *Denote the feature correlation matrices of online and target outputs as* $\mathbb{C}_p = \mathbb{E}_x p_x p_x^\top$ *and* $\mathbb{C}_z = \mathbb{E}_x z_x z_x^\top$*. During training, the alignment loss will align the eigenspace of* $\mathbb{C}_p$ *and* $\mathbb{C}_z$*,* $\exists\, V$*, s.t.* $\mathbb{C}_p = V\Lambda_p V^\top, \mathbb{C}_z = V\Lambda_z V^\top$*, where* Λ_p, Λ_z *are two diagonal matrices consisting of the eigenvalues* $\lambda_i^p, \lambda_i^z, i = 1, \dots, k$*, respectively.*

This mechanism is a natural consequence of the alignment loss that enforces the two branches to produce nearly the same features. Tian et al. (2021) provide a theoretical justification of eigenspace alignment for predictor-based methods under simplified linear scenarios. Nevertheless, it is still hard to provide general justifications for practical scenarios with nonlinear models and various asymmetric designs. Instead, we provide an empirical examination of different non-contrastive methods. From Figure 3, we can see that there is a consistent high degree of eigenspace alignment among all methods.[2] Therefore, these methods indeed attain a fairly high degree of eigenspace alignment.

We note that Mechanism 1 applies to both symmetric and asymmetric designs due to the common alignment loss. As a result of eigenspace alignment, the alignment loss essentially works on mitigating the difference in *eigenvalues*. Therefore, we can take a spectral perspective on non-contrastive methods, where the asymmetric designs are *equivalent* to an online spectral filter $g(\lambda^z) = \sqrt{\lambda^p/\lambda^z}$ applied to the target output z_x, or equivalently, a target spectral filter $h(\lambda^p) = \sqrt{\lambda^z/\lambda^p}$ applied to the target output p_x, as formally characterized by Lemma 2. As the two cases are equivalent, we mainly take the online filter as an example in the discussion below.

Lemma 2. *Given a filter function* $g(\lambda_i^z) = \sqrt{\lambda_i^p/\lambda_i^z}$*, we can apply a linear transformation* $W_g = Vg(\Lambda_z)V^\top$ *on* z_x *and get* $\tilde{p}_x = W_g z_x$*, where* $g(\Lambda_z)$ *is a diagonal spectral filter with* $g(\Lambda_z)_{ii} = g(\lambda_i^z)$*. Denoting the correlation matrix of* $\tilde{p}_x$ *as* $\mathbb{C}_{\tilde{p}} = \mathbb{E}_x \tilde{p}_x \tilde{p}_x^\top$*, we have* $\mathbb{C}_{\tilde{p}} = \mathbb{C}_p = \mathbb{E}_x p_x p_x^\top$*.*

This spectral view reveals the key difference between symmetric and asymmetric designs. In the symmetric case, the two branches yield nearly equal eigenvalues (Figure 2(d)). Thus, the alignment loss will quickly diminish and lead to dimensional collapse. Instead, in asymmetric designs, we find that the architectural asymmetry intentionally creates a difference in the eigenvalues such that the target output generally has higher rank than the online output, as revealed in the next mechanism. In this case, unlike symmetric ones, the alignment loss will not easily diminish, and even not necessarily consistently decrease, as observed in practical non-contrastive learning (Figure 9).

Mechanism 2 (Rank Difference)**.** *The asymmetric designs in non-contrastive methods behave as a low-pass filter on the online branch, i.e.,* $g(\lambda_i^z) = \sqrt{\lambda_i^p/\lambda_i^z}$ *that is monotonically increasing, or equivalently, a high-pass filter on the target branch, i.e.,* $h(\lambda_i^p) = \sqrt{\lambda_i^z/\lambda_i^p}$ *that is monotonically decreasing. This filtering effect yields a consistent rank difference between the online and target branches, i.e.,* $\mathrm{erank}(\mathbb{C}_p) < \mathrm{erank}(\mathbb{C}_z)$*, as long as the filter is non-constant.*

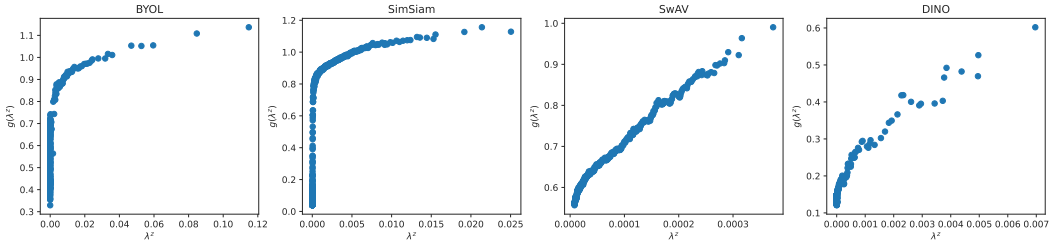
From Figure 4(a), we can see a consistent difference between online and target branches among different methods. The distribution of eigenvalues is more concentrated (*i.e.,* less uniform) on the online branch. As a result, the effective rank of the online output will be smaller than that of the target output, which explains the positive rank difference in Figure 1. We further investigate the spectral filter $g(\lambda^z)$ empirically chosen or learned by each non-contrastive method. From Figure 4(b), we find that among all methods, the online filter $g(\lambda_i^z)$ is nearly monotonically increasing in λ_i^z. In other words, $g(\lambda^z)$ is a low-pass filter that, relatively speaking, enlarges the low-frequency components with large eigenvalues, and shrinks the high-frequency components with small eigenvalues. The following theorem proves that such a low-pass filter will indeed create a consistent rank difference. In Section 5, we provide further theoretical understandings for why they behave as low-pass filters.

Theorem 3. *If* $g(\lambda_i^z) = \sqrt{\lambda_i^p/\lambda_i^z}$ *is monotonically increasing, or equivalently,* $h(\lambda_i^p) = \sqrt{\lambda_i^z/\lambda_i^p}$ *is monotonically decreasing, we have* $\mathrm{erank}(\mathbb{C}_p) \leq \mathrm{erank}(\mathbb{C}_z)$*. Further, as long as* $g(\lambda_i^z)$ *or* $h(\lambda_i^p)$ *is non-constant, the inequality holds strictly,* $\mathrm{erank}(\mathbb{C}_p) < \mathrm{erank}(\mathbb{C}_z)$*.*

Indeed, as shown in Figure 1, the effective rank of the target output is consistently larger than the online output during the training process. Meanwhile, we also observe that for each output, its own

[2] As DINO adopts dynamic feature sharpening coefficients, there is a larger change of its eigenspace alignment compared to others. Nevertheless, the alignment degree is always above 0.7, which is relatively high.

(a) Eigenvalues of the feature correction matrices of the online and target outputs.



(b) The corresponding online spectral filter function $g(\lambda_i^z) = \sqrt{\lambda_i^p / \lambda_i^z}$.

Figure 4: Rank difference experiments and spectral filters on CIFAR-10. (a): Comparison of the eigenvalues computed from two branch outputs. There is a clear difference in eigenvalues, and the eigenvalue distribution of the online branch is more biased towards larger values. (b): The spectral filter $g(\lambda_i^z) = \sqrt{\lambda_i^p / \lambda_i^z}$, where λ_i^p, λ_i^z are eigenvalues of online and target correlation matrices $\mathbb{C}_p, \mathbb{C}_z$, respectively. The spectral filters of all methods are nearly monotonically increasing *w.r.t.* λ_i^z.

effective rank is also successfully elevated along this process. This is not a coincidence. Below, we will show how the rank difference helps alleviate dimensional collapse while training.

Mechanism 3 (Rank Improvement)**.** *With eigenspace alignment (Mechanism 1) and rank difference (Mechanism 2), the alignment between the online and target outputs will improve their own effective feature dimensionality and alleviate dimensional collapse.*

The improvement of effective rank is a natural consequence of the rank difference. Intuitively, as the online output has lower effective rank than the target output, optimizing its alignment loss *w.r.t.* the gradient-detached target $\mathrm{sg}(z_x)$ will enforce the online output p_x to improve its effective rank in order to match the target output z_x. In this way, the rank difference becomes a ladder (created by asymmetric designs) for non-contrastive methods to gradually improve its effective feature dimensionality and get rid of dimensional collapse eventually.

Nevertheless, a formal exposure of Mechanism 3 requires examining the training dynamics of deep neural networks, which are generally hard to analyze. Prior works (Tian et al., 2021; Wen & Li, 2021) mainly deal with linear or shadow networks with strong assumptions on data distribution, which could be far from practice. As over-parameterized deep neural networks are very expressive, we instead adopt the unconstrained feature setting (Mixon et al., 2022; HaoChen et al., 2021) and consider gradient descent directly in the feature space $\mathbb{R}^k$. Taking the MSE loss $\ell(u, v) = \frac{1}{2}\|u - v\|^2$ as an example, in the following theorem, we provide a theoretical justification of Mechanism 3 by showing that the rank difference indeed helps improve the effective rank of the online output p.

Theorem 4. *Assuming the eigenspace alignment of the online and target outputs, the gradient descent update of online features at the t-th step with step size $0 < \alpha < 1$ is*

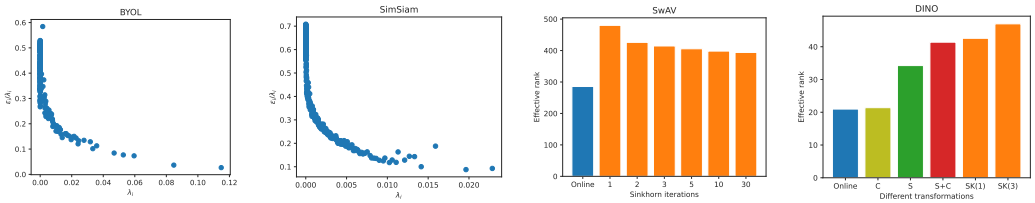
$$p^{(t+1)} = p^{(t)} - \alpha \nabla_p \mathcal{L}(p) = p^{(t)} - \alpha(p^{(t)} - z^{(t)}), \tag{3}$$

which gives the following update on feature correlation matrix $\mathbb{C}_p$ element-wisely,

$$\lambda_{i,t+1}^p = \lambda_{i,t}^p + 2\alpha\sqrt{\lambda_{i,t}^p}\left(\sqrt{\lambda_{i,t}^z} - \sqrt{\lambda_{i,t}^p}\right) + \alpha^2\left(\sqrt{\lambda_{i,t}^z} - \sqrt{\lambda_{i,t}^p}\right)^2, i = 1 \ldots, k. \tag{4}$$

The corresponding spectral filter $g_t(\lambda_{i,t}^p) = \lambda_{i,t+1}^p / \lambda_{i,t}^p$ is monotonically decreasing w.r.t. $\lambda_{i,t}^p$ and non-constant. Then, according to Theorem 3, we have $\mathrm{erank}(\mathbb{C}_{p^{(t+1)}}) > \mathrm{erank}(\mathbb{C}_{p^{(t)}})$.

Remark on Stop Gradient. Our analysis above also reveals the importance of the stop gradient operation. In particular, when gradients from the target branch are not detached, in order to match



(a) BYOL (b) SimSiam (c) SwAV (d) DINO

Figure 5: The role of asymmetric designs. (a) & (b): The corresponding spectral filters of data augmentations, ϵ_i/λ_i v.s. λ_i. Both filters are nearly monotonically decreasing with respect to λ_i. (c): The effective rank of the outputs of SwAV with different Sinkhorn-Knopp (SK) iterations. (d): The effective rank of the outputs of DINO with different operations. C: Centering, S: Sharpening, C+S: Centering and Sharpening, SK(i): Sinkhorn-Knopp iteration with i steps.

the rank of two outputs, minimizing the alignment loss may also *pull down* the rank of the target output. As a result, the feature rank can hardly be consistently improved without stop gradient.

5 How to Obtain Rank Difference with Asymmetric Designs: Existing Techniques and Beyond

In Section 4, we have shown that the essential mechanism of non-contrastive methods for alleviating dimensional collapse is to create a rank difference between online and target branches. In this section, we provide some further theoretical and empirical insights on how existing asymmetric designs could realize the rank difference. Besides, our rank differential mechanism also provides guidelines for designing new asymmetry variants that also work effectively in practice.

5.1 Existing Technique I: Learnable Online Predictor

One kind of non-contrastive methods, including BYOL and SimSiam, utilizes a learnable online predictor $g_\theta : \mathbb{R}^k \to \mathbb{R}^k$ for architectural asymmetry. Taking the simpler SimSiam as an example, it aligns the positive pair with the asymmetric alignment loss, $\mathcal{L}(\theta) = \mathbb{E}_{x,x^+}\frac{1}{2}\|g_\theta(f_\theta(x)) - \mathrm{sg}(f_\theta(x^+))\|^2$. Figure 4(a) shows that this online predictor behaves like a low-pass filter, which is critical to producing the rank difference as desired. In turn, one would wonder why such a *learnable* predictor will behave as a low-pass filter (Lemma 2). Here, we provide some theoretical insights into this phenomenon in the following theorem. For simplicity, we consider a linear predictor $g_\theta(z) = Wz, W \in \mathbb{R}^{k\times k}$, whose alignment loss is $\mathcal{L}(W) = \mathbb{E}_{x,x^+}\frac{1}{2}\|Wz_x - \mathrm{sg}(z_{x^+})\|^2$.

Theorem 5. *Denote the feature correlation matrix of natural data $\bar{x}$ as $\mathbb{C}_{\bar{z}} = \mathbb{E}_{\bar{x}} z_{\bar{x}} z_{\bar{x}}^\top$, where its representation $z_{\bar{x}} = \mathbb{E}_{x|\bar{x}} z_x$ is computed by average over augmentations $x \sim p(x|\bar{x})$. Assuming that $\mathbb{C}_z, \mathbb{C}_{\bar{z}}$ have aligned eigenspace V, we have the following conclusions:*

1. *We have $\mathbb{C}_z = \mathbb{C}_{\bar{z}} + \mathbb{V}_{z|\bar{z}}$, where $\mathbb{V}_{z|\bar{z}} = \mathbb{E}_{\bar{x}}\left[\mathbb{E}_{x|\bar{x}}\left(z_x - z_{\bar{x}}\right)\left(z_x - z_{\bar{x}}\right)^\top\right]$ is the conditional feature covariance matrix introduced by data augmentation. The corresponding eigenvalues satisfy $\lambda_i^z = \lambda_i^{\bar{z}} + \varepsilon_i$, where $\bar{\lambda}_i, \varepsilon_i \geq 0$ denote the i-th eigenvalues of $\mathbb{C}_{\bar{z}}, \mathbb{V}_{z|\bar{z}}$, respectively.*
2. *With an invertible C_z, the optimal predictor weight W^* is given by*

$$W^* = \mathbb{C}_{\bar{z}}\mathbb{C}_z^{-1} = V\Omega V^\top, \quad \text{where } \omega_i = \Omega_{ii} = \frac{\bar{\lambda}_i}{\lambda_i} = \frac{\lambda_i - \varepsilon_i}{\lambda_i} = 1 - \frac{\varepsilon_i}{\lambda_i} \in [0, 1]. \tag{5}$$

Therefore, when $g_a(\lambda_i) = \varepsilon_i/\lambda_i$ is monotonically decreasing, w_i is monotonically increasing. In other words, $g(z) = Wz$ is a low-pass filter.

Theorem 5 shows that the optimal predictor W is determined by the eigenvalue ratio $\omega_i = \bar{\lambda}_i/\lambda_i$ between the features of natural and augmented data, and their difference $\varepsilon_i = \lambda_i - \bar{\lambda}_i$ is induced by the randomness in data augmentations $p(x|\bar{x})$. In Figure 5(a) and 5(b), we examine the filter ε_i/λ_i to study the effect of data augmentation. We notice that ε_i/λ_i is a high-pass filter as it is nearly monotonically decreasing, which suggests that data augmentation mainly contributes to the *high-frequency* noise in the feature z_x. In this case, because the predictor $W^* = \mathbb{C}_{\bar{z}}\mathbb{C}_z^{-1}$ is learned to

Table 1: Linear probing accuracy (%) of SimSiam with different online predictors on CIFAR-10, CIFAR-100 and ImageNet-100.

Predictor	CIFAR-10			CIFAR-100			ImageNet-100 (400ep)	
	100ep	200ep	400ep	100ep	200ep	400ep	acc@1	acc@5
Learnable (nonlinear)	74.00	84.39	89.41	46.12	55.51	63.39	77.96	94.26
Learnable (linear)	72.42	**83.67**	**88.9**	42.98	**55.69**	**62.02**	77.18	94.02
$g(\sigma) = \sigma$	73.91	82.74	86.51	**46.73**	54.45	59.59	75.06	93.28
$g(\sigma) = \log(\sigma)$	74.80	80.72	87.52	39.82	48.59	58.08	77.22	**94.34**
$g(\sigma) = \log(1+\sigma)$	**75.70**	80.47	86.76	40.53	47.68	61.15	**77.42**	93.90
$g(\sigma) = \log(1+\sigma^2)$	75.51	82.28	87.51	39.19	48.97	59.06	77.22	94.00

Table 2: Linear probing accuracy (%) of SimSiam with different target predictors on CIFAR-10, CIFAR-100 and ImageNet-100.

Predictor	CIFAR-10			CIFAR-100			ImageNet-100 (400ep)	
	100ep	200ep	400ep	100ep	200ep	400ep	acc@1	acc@5
Learnable (nonlinear)	74.00	84.39	89.41	46.12	55.51	63.39	77.96	94.26
Learnable (linear)	72.42	83.67	**88.90**	42.98	55.69	62.02	77.18	94.02
$h(\sigma) = \sigma^{-0.3}$	77.33	85.57	88.74	48.07	58.07	**62.83**	**77.68**	94.16
$h(\sigma) = \sigma^{-0.5}$	**78.17**	85.52	88.43	50.61	58.68	62.79	76.74	94.10
$h(\sigma) = \sigma^{-0.7}$	77.98	**85.79**	88.45	**51.04**	58.37	62.69	77.28	**94.94**
$h(\sigma) = \sigma^{-1}$	77.33	86.44	88.50	51.05	**58.87**	61.92	77.00	93.32

reconstruct the natural feature $z_{\bar{x}}$ from the augmented one z_x, it has to eliminate the high-frequency noise with a converse filter $1 - \varepsilon_i/\lambda_i$. As a result, the learnable predictor behaves as a low-pass filter that could introduce the rank difference and alleviate dimensional collapse (Section 4.2).

5.2 Existing Technique II: Target Feature Transformation

Another kind of non-contrastive method achieves asymmetry by applying additional feature transformations on the target branch, *e.g.,* SwAV and DINO. Specifically, SwAV applies a few steps of Sinkhorn-Knopp (SK) iteration (Cuturi, 2013), while DINO combines feature centering and feature sharpening. In previous discussions, both two methods are shown to produce a high-pass target filter $h(\lambda_i^p) = \sqrt{\lambda_i^z/\lambda_i^p}$ (Figure 4(b)) and the effective rank of the target output is consistently higher than the online output (Figure 1).

Since these feature transformations are directly operating in the spatial domain, they typically do not have a direct correspondence in the spectral domain, making it hard for formal analysis. To gain some further insights, we investigate the effect of each transformation on the effective rank. Figure 5(c) shows that SK iteration will indeed improve the effective rank, while longer iterations do not necessarily contribute to a larger effective rank. This explains why SwAV only adopts a few iterations. As for DINO, we study several asymmetric operations considered in DINO: feature centering (C), feature sharpening (S), feature centering and sharpening (C+S, default in DINO), and SK iteration with i steps (SK(i)). Figure 5(d) shows that feature sharpening (instead of feature centering) is the key for creating rank differences, while combining the two can lead to further improvement and reach similar rank differences as SK iteration, which also aligns well with the empirical performance of these techniques (see Appendix B & D in Caron et al. (2021)).

5.3 Inspired New Methods

The discussions above show that the rank differential mechanism provides a unified theoretical understanding of different non-contrastive methods. Besides, it also provides a general recipe for designing *new* non-contrastive variants. As Theorem 3 points out, the key requirement is that the asymmetry can produce a low-pass filter on the online branch, or equivalently, a high-pass filter on the target branch. Below, we propose some new variants directly following this principle.

Variants of Online Low-pass Filters. Despite the learnable predictor (Section 5.1), we can also directly design online predictors with fixed low-pass filters. For numerical stability, we adopt the singular value decomposition (SVD) of the output to compute the eigenvalues and the eigenspace, *e.g.,* $z = U\Sigma_z V^\top$ with singular values σ_i^z's. As $\lambda_i^z = (\sigma^{z_i})^2$, a filter that is monotone in λ is also

Table 3: Linear probing accuracy (%) of SimSiam with different predictors on ImageNet-1k.

SimSiam (Learnable Online Filter)		Online Low-pass Filter (ours) $g(\sigma) = \log(1+\sigma)$		Target High-pass Filter (ours) $g(\sigma) = \sigma^{-0.3}$	
acc@1	acc@5	acc@1	acc@5	acc@1	acc@5
67.97	88.17	64.67	86.13	67.73	88.05

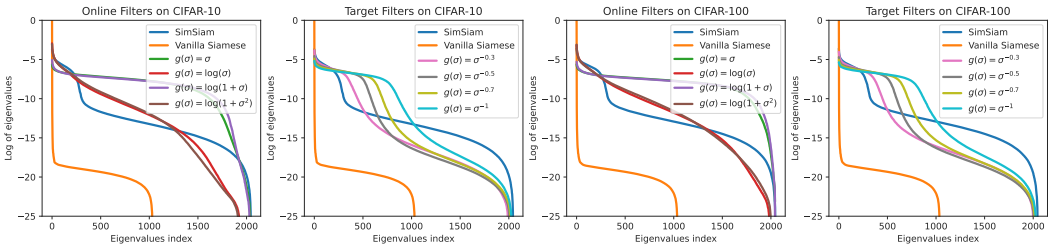


Figure 6: Eigenvalues of the correlation matrix of the normalized outputs for different variants of filters on CIFAR-10 and CIFAR-100.

monotone in σ, and vice versa. Specifically, for an online encoder $f : \mathcal{X} \to \mathbb{R}^k$, we assign $W = Vg(\Sigma^z)V^\top$, where $g(\Sigma^z)_{ii} = g(\sigma_i^z)$ is a low-pass filter that is monotonically increasing *w.r.t.* σ. We note that DirectPred proposed by Tian et al. (2021) is a special case with $g_0(\sigma) = \sigma$. Additionally, we consider three variants: 1) $g_1(\sigma) = \log(\sigma)$; 2) $g_2(\lambda) = \log(\sigma + 1)$; 3) $g_3(\sigma) = \log(\sigma^2 + 1)$. These three new variants are low-pass filters because they are monotonically increasing with $\sigma \geq 0$.

We evaluate their performance on four datasets, *i.e.,* CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), ImageNet-100 and ImageNet-1k (Deng et al., 2009). We use ResNet-18 (He et al., 2016) as the backbone encoder for CIFAR-10, CIFAR-100 and ImageNet-100 and adopt ResNet-50 for Imagenet-1k following standard practice. And the projector is a three-layer MLP which BN (Ioffe & Szegedy, 2015) is applied to all layers. We adopt the linear probing task for evaluating the learned representations. More details are included in Appendix B.2. Table 1 shows that our designs of online predictors with different low-pass filters work well in practice, and achieve comparable performance to SimSiam with learnable predictors. In particular, it also significantly outperforms SymSimSiam (Figure 2(a)), showing that rank differences indeed helps alleviate dimensional collapse (Figure 6).

Variants of Target High-pass Filters. In turn, we also consider applying a high-pass filter on the target branch using a *target predictor*. Compared to online predictors above, target predictors have additional advantages: with stop gradient, we do not require backpropagation through SVD, which could reduce time overhead (Table 5). Specifically, we consider polynomial high-pass target filters $h(\sigma) = \sigma^p$ with different $-1 \leq p < 0$, which are all monotonically decreasing (see Algorithm 2).

We further evaluate the target filters following the same protocols above. As shown in Table 2, our high-pass target filters can often outperform SimSiam with a large margin with relatively short training time. Especially, on CIFAR-10, the filter $h(\sigma) = \sigma^{-0.5}$ improves the baseline by 5.75% with 100 epochs and the filter $h(\sigma) = \sigma^{-0.7}$ improves the baseline by 2.12% with 200 epochs. Notably, on CIFAR-100, our methods outperform the baseline by a large margin (8.06%, 3.18%, and 0.81% improvements with 100, 200, and 400 epochs, respectively). Additional results based on BYOL framework can be found in Appendix C.

6 Conclusion

In this paper, we presented a unified theoretical understanding of non-contrastive learning via the rank differential mechanism. In particular, we showed that existing non-contrastive learning all produce a consistent rank difference between the online and the target outputs. Digging deeper into this phenomenon, we theoretically proved that low-pass online filters can yield such a rank difference and improve the effective feature dimensionality along training. Meanwhile, we provided theoretical and empirical insights on how existing asymmetric designs produce low-pass filters. At last, following the principle of our theory, we designed a series of new online and target filters, and showed that they achieve comparable or even superior to existing asymmetric designs.

REFERENCES

Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Dipendra Misra. Investigating the role of negatives in contrastive representation learning. In *AISTATS*, 2022.

Han Bao, Yoshihiro Nagano, and Kento Nozawa. On the surrogate gap between contrastive and supervised losses. In *ICML*, 2022.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013.

Victor Guilherme Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. solo-learn: A library of self-supervised methods for visual representation learning. *Journal of Machine Learning Research*, 23:56–1, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In *ICML*, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altché, C. Tallec, Pierre H. Richemond, Elena Buchatskaya, C. Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, B. Piot, K. Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: a new approach to self-supervised learning. In *NeurIPS*, 2020.

Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. In *NeurIPS*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Tianyu Hua, Wenxiao Wang, Zihui Xue, Sucheng Ren, Yue Wang, and Hang Zhao. On feature decorrelation in self-supervised learning. In *ICCV*, 2021.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *ICLR*, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Jason D Lee, Qi Lei, Nikunj Saunshi, and Jiacheng Zhuo. Predicting what you already know helps: Provable self-supervised learning. In *NeurIPS*, 2021.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

Dustin G Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features. *Sampling Theory, Signal Processing, and Data Analysis*, 20(2):1–13, 2022.

Kento Nozawa and Issei Sato. Understanding negative samples in instance discriminative self-supervised representation learning. In *NeurIPS*, 2021.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *EUSIPCO*, 2007.

Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. A theoretical analysis of contrastive unsupervised representation learning. In *ICML*, 2019.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *ICML*, 2021.

Christopher Tosh, Akshay Krishnamurthy, and Daniel Hsu. Contrastive learning, multi-view redundancy, and linear models. In *COLT*, 2021.

Xiao Wang, Haoqi Fan, Yuandong Tian, Daisuke Kihara, and Xinlei Chen. On the importance of asymmetry for siamese representation learning. In *CVPR*, 2022a.

Yifei Wang, Qi Zhang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. Chaos is a ladder: A new theoretical understanding of contrastive learning via augmentation overlap. In *ICLR*, 2022b.

Zixin Wen and Yuanzhi Li. Toward understanding the feature learning process of self-supervised contrastive learning. In *ICML*, 2021.

Zixin Wen and Yuanzhi Li. The mechanism of prediction head in non-contrastive self-supervised learning. *arXiv preprint arXiv:2205.06226*, 2022.

Yang You, Igor Gitman, and Boris Ginsburg. Scaling SGD batch size to 32k for ImageNet training. *arXiv preprint arXiv:1708.03888*, 2017.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

Chaoning Zhang, Kang Zhang, Chenshuang Zhang, Trung X Pham, Chang D Yoo, and In So Kweon. How does simsiam avoid collapse without negative samples? a unified understanding with self-supervised contrastive learning. In *ICLR*, 2022.

A OMITTED PROOFS

In this section, we present proofs for all lemmas and theorems in the main paper.

A.1 PROOF OF THEOREM 1

Proof. Since the output of f_θ is ℓ_2-normalized, and $\mathbb{E}_x f(x) = \mathbb{E}_{x^+} f(x^+) = \mu$, we have $\mathcal{L}_{\text{sym}}(\theta) = -\mathbb{E}_{x,x^+} f'_\theta(x)^\top f'_\theta(x^+) = -\mathbb{E}_{x,x^+}(f_\theta(x) - \mu)^\top (f_\theta(x^+) - \mu) = -\mathbb{E}_{x,x^+} f_\theta(x)^\top f_\theta(x^+) + \|\mu\|^2$, and $\text{Var}(f(x)) = \mathbb{E}_x \|f(x)\|^2 - \|\mu\|^2 = 1 - \|\mu\|^2$. Thus, we conclude that $\mathcal{L}_{\text{sym}}(\theta) = -\mathbb{E}_{x,x^+} f_\theta(x)^\top f_\theta(x^+) - \text{Var}(f(x)) + 1$. □

A.2 PROOF OF LEMMA 2

Proof. By the definition of $\tilde{p}_x = W_g z_x$, we know that

$$\mathbb{C}_{\tilde{p}} = \mathbb{E}_x \tilde{p}_x \tilde{p}_x^\top = \mathbb{E}_x W_g z_x z_x^\top W_g^\top = W_g \left(\mathbb{E}_x z_x z_x^\top\right) W_g^\top = W_g \mathbb{C}_z W_g^\top. \tag{6}$$

Since $W_g = V g(\Lambda_z) V^\top, \mathbb{C}_z = V \Lambda_z V^\top, \mathbb{C}_p = V \Lambda_p V^\top$ and $g(\lambda_i^z) = \sqrt{\lambda_i^p / \lambda_i^z}$, we have

$$\mathbb{C}_{\tilde{p}} = V g(\Lambda_z) V^\top V \Lambda_z V^\top V g(\Lambda_z) V^\top = V g(\Lambda_z) \Lambda_z g(\Lambda_z) V^\top = V \Lambda_p V^\top = \mathbb{C}_p. \tag{7}$$

□

A.3 PROOF OF THEOREM 3

Before the proof of Theorem 3, we first introduce the following useful lemma.

Lemma 6. *Assume that $\sum_{i=1}^k q_i = 1$ and $q_1 \geq q_2 \geq \cdots \geq q_k > 0$, then for any $1 \leq i < j \leq k$ and any $\Delta \in (0, p_j)$, it holds that*

$$H(q_1, q_2, \ldots, q_k) > H(q_1, \ldots, q_i + \Delta, \ldots, q_j - \Delta, \ldots, q_k). \tag{8}$$

Proof. We first note that

$$H(q_1, q_2, \ldots, q_k) > H(q_1, \ldots, q_i + \Delta, \ldots, q_j - \Delta, \ldots, q_k) \tag{9}$$

$$\iff -q_i \log(q_i) - q_j \log(q_j) > -(q_i + \Delta)\log(q_i + \Delta) - (q_j - \Delta)\log(q_j - \Delta). \tag{10}$$

Define $f(x) = -(q_i + x)\log(q_i + x) - (q_j - x)\log(q_j - x)$ and its first order derivative satisfies

$$f'(x) = -\log(q_i + x) + \log(q_j - x) < 0, \quad \forall x \in (0, q_j). \tag{11}$$

Hence, Equation (10) holds. This completes the proof of the lemma. □

Proof of Theorem 3. Let $q_i^z = \lambda_i^z / (\sum_{l=1}^k \lambda_l^z)$ and $q_i^p = \lambda_i^p / (\sum_{l=1}^k \lambda_l^p)$, where $i = 1, 2, \ldots, k$, then $\sum_{i=1}^k q_i^z = \sum_{i=1}^k q_i^p = 1$, $q_1^z \geq q_2^z \geq \cdots \geq q_k^z$ and $q_1^z \geq q_2^z \geq \cdots \geq q_k^z$. Without loss of generality, we assume that $q_k^p, q_k^z > 0$. Because $g(\lambda_i^z) = \sqrt{\lambda_i^p / \lambda_i^z}$ is monotonically increasing and $\lambda_i^z \geq \lambda_j^z$, for any $1 \leq i < j \leq k$, we have

$$\frac{q_i^p}{q_j^p} = \frac{\lambda_i^p / (\sum_{l=1}^k \lambda_l^p)}{\lambda_j^p / (\sum_{l=1}^k \lambda_l^p)} = \frac{\lambda_i^p}{\lambda_j^p} = \frac{g^2(\lambda_i^z)\lambda_i^z}{g^2(\lambda_j^z)\lambda_j^z} \geq \frac{\lambda_i^z}{\lambda_j^z} = \frac{\lambda_i^z / (\sum_{l=1}^k \lambda_l^z)}{\lambda_j^z / (\sum_{l=1}^k \lambda_l^z)} = \frac{q_i^z}{q_j^z}. \tag{12}$$

If $g(\lambda_i^z)$ is constant, it follows that

$$\frac{q_i^p}{q_j^p} = \frac{q_i^z}{q_j^z}, \quad \forall 1 \leq i < j \leq k. \tag{13}$$

Combining with $\sum_{i=1}^k q_i^z = \sum_{i=1}^k q_i^p = 1$, we get $q_i^p = q_i^z, i = 1, 2, \ldots, k$. Hence,

$$H(q_1^p, q_2^p, \ldots, q_k^p) = H(q_1^z, q_2^z, \ldots, q_k^z), \tag{14}$$

which implies that $\text{erank}(\mathbb{C}_p) = \text{erank}(\mathbb{C}_z)$.

If $g(\lambda_i^z)$ is non-constant, then $g(\lambda_1^z) > g(\lambda_k^z)$. And it follows that

$$\frac{q_1^p}{q_k^p} < \frac{q_1^z}{q_k^z}. \tag{15}$$

Arming with Equations (12) and (15), we get

$$1 = \sum_{i=1}^{k} q_i^p = q_k^p \sum_{i=1}^{k} \frac{q_i^p}{q_k^p} > q_k^p \sum_{i=1}^{k} \frac{q_i^z}{q_k^z} = \frac{q_k^p}{q_k^z} \sum_{i=1}^{k} q_i^z = \frac{q_k^p}{q_k^z}, \tag{16}$$

which indicates $q_k^p < q_k^z$. Similarly, we have $q_1^p > q_1^z$. Hence, $m = \max\{i | q_i^p \geq q_i^z, i = 1, 2, \ldots, k\}$ exists and $m < 1$. Using Equation (12), we have

$$\begin{cases} q_i^p \geq q_i^z & \text{if } 1 \leq i \leq m, \\ q_i^p < q_i^z & \text{if } m+1 \leq i \leq k. \end{cases} \tag{17}$$

Directly applying $\sum_{i=1}^{k} q_i^z = \sum_{i=1}^{k} q_i^p = 1$ gives that

$$\sum_{i=1}^{m} \underbrace{q_i^p - q_i^z}_{\geq 0} = \sum_{i=m+1}^{k} \underbrace{q_i^z - q_i^p}_{>0}. \tag{18}$$

According to Lemma 6 and Equation (18), if we transport the redundancy from right to left, the entropy of the distribution will decrease. The transportation process is described as following:

Step 1 Let $i \leftarrow 1$ and $j \leftarrow k$.

Step 2 If $\Delta = \min\{q_i^p - q_i^z, q_j^z - q_j^p\} > 0$, then $q_i^z \leftarrow q_i^z + \Delta$ and $q_j^z \leftarrow q_j^z - \Delta$.

Step 3 If $q_i^p = q_i^z$, $i \leftarrow i + 1$. Else $j \leftarrow j - 1$

Step 4 If $i \geq j$, we finish this process. Else, we return step 2.

After at most $k - 1$ loops, $(q_1^z, q_2^z, \ldots, q_k^z)$ becomes $(q_1^p, q_2^p, \ldots, q_k^p)$. Equation (18) ensures the correctness of the transportation process. According to Lemma 6, we have

$$H(q_1^z, q_2^z, \ldots, q_k^z) > H(q_1^p, q_2^p, \ldots, q_k^p), \tag{19}$$

which implies that $\mathrm{erank}(\mathbb{C}_p) < \mathrm{erank}(\mathbb{C}_z)$.

□

A.4 Proof of Theorem 4

Proof. Since $p^{(t+1)} = p^{(t)} - \alpha(p^{(t)} - z^{(t)}) = (1-\alpha)p^{(t)} + \alpha z^{(t)}$,

$$\left(p^{(t+1)}\right)^\top p^{(t+1)} = \left[(1-\alpha)p^{(t)} + \alpha z^{(t)}\right]^\top \left[(1-\alpha)p^{(t)} + \alpha z^{(t)}\right] \tag{20}$$

$$= (1-\alpha)^2 \left(p^{(t)}\right)^\top p^{(t)} + \alpha^2 \left(z^{(t)}\right)^\top z^{(t)}$$

$$+ \alpha(1-\alpha)\left[\left(p^{(t)}\right)^\top z^{(t)} + \left(z^{(t)}\right)^\top p^{(t)}\right]. \tag{21}$$

It follows that

$$V\Lambda_p^{(t+1)}V^\top = (1-\alpha)^2 V\Lambda_p^{(t)}V^\top + 2\alpha(1-\alpha)V\left(\Lambda_p^{(t)}\Lambda_z^{(t)}\right)^{1/2}V^\top + \alpha^2 V\Lambda_z^{(t)}V^\top. \tag{22}$$

As a result, we have

$$\Lambda_p^{(t+1)} = (1-\alpha)^2\Lambda_p^{(t)} + 2\alpha(1-\alpha)\left(\Lambda_p^{(t)}\Lambda_z^{(t)}\right)^{1/2} + \alpha^2\Lambda_z^{(t)}. \tag{23}$$

Therefore,

$$\lambda_{i,t+1}^p = (1-\alpha)^2\lambda_{i,t}^p + 2\alpha(1-\alpha)\sqrt{\lambda_{i,t}^p\lambda_{i,t}^z} + \alpha^2\lambda_{i,t}^z \tag{24}$$

$$= \lambda_{i,t}^p + 2\alpha(\sqrt{\lambda_{i,t}^p\lambda_{i,t}^z} - \lambda_{i,t}^p) + \alpha^2\left(\lambda_{i,t}^z - 2\sqrt{\lambda_{i,t}^p\lambda_{i,t}^z} + \lambda_{i,t}^p\right) \tag{25}$$

$$= \lambda_{i,t}^p + 2\alpha\sqrt{\lambda_{i,t}^p}\left(\sqrt{\lambda_{i,t}^z} - \sqrt{\lambda_{i,t}^p}\right) + \alpha^2\left(\sqrt{\lambda_{i,t}^z} - \sqrt{\lambda_{i,t}^p}\right)^2, i = 1\ldots, k. \tag{26}$$

Dividing both sides of Equation (25 by $\lambda_{i,t}^p$, we have

$$\frac{\lambda_{i,t+1}^p}{\lambda_{i,t}^p} = (1-\alpha)^2 + 2\alpha(1-\alpha)\sqrt{\frac{\lambda_{i,t}^z}{\lambda_{i,t}^p}} + \alpha^2 \frac{\lambda_{i,t}^z}{\lambda_{i,t}^p}. \tag{27}$$

Let $g_t(\lambda_{i,t}^p) = \frac{\lambda_{i,t+1}^p}{\lambda_{i,t}^p}$ and $h_t(\lambda_{i,t}^p) = \sqrt{\frac{\lambda_{i,t}^z}{\lambda_{i,t}^p}}$. Since $h_t(\lambda_{i,t}^p)$ is monotonically decreasing and non-constant and $0 < \alpha < 1$, $g_t(\lambda_{i,t}^p) = (1-\alpha)^2 + 2\alpha(1-\alpha)h_t(\lambda_{i,t}^p) + \alpha^2 h_t^2(\lambda_{i,t}^p)$ is also monotonically decreasing and non-constant. According to Theorem 2, we have $\operatorname{erank}(\mathbb{C}_{p^{(t+1)}}) > \operatorname{erank}(\mathbb{C}_{p^{(t)}})$.

□

A.5 Proof of Theorem 5

Proof. For the first part,

$$\begin{aligned}
\mathbb{C}_{\bar{z}} + \mathbb{V}_{z|\bar{z}} &= \mathbb{E}_{\bar{x}} z_{\bar{x}} z_{\bar{x}}^\top + \mathbb{E}_{\bar{x}}\left[\mathbb{E}_{x|\bar{x}}\left(z_x - z_{\bar{x}}\right)\left(z_x - z_{\bar{x}}\right)^\top\right] && (28)\\
&= \mathbb{E}_{\bar{x}} z_{\bar{x}} z_{\bar{x}}^\top + \mathbb{E}_{\bar{x}}\left[\mathbb{E}_{x|\bar{x}} z_x z_x^\top - \left(\mathbb{E}_{x|\bar{x}} z_x\right) z_{\bar{x}}^\top - z_{\bar{x}}\left(\mathbb{E}_{x|\bar{x}} z_x\right)^\top + z_{\bar{x}} z_{\bar{x}}^\top\right] && (29)\\
&= \mathbb{E}_{\bar{x}} z_{\bar{x}} z_{\bar{x}}^\top + \mathbb{E}_{\bar{x}}\mathbb{E}_{x|\bar{x}} z_x z_x^\top - \mathbb{E}_{\bar{x}} z_{\bar{x}} z_{\bar{x}}^\top && (30)\\
&= \mathbb{E}_x z_x z_x^\top && (31)\\
&= \mathbb{C}_z. && (32)
\end{aligned}$$

Since $\mathbb{C}_z, \mathbb{C}_{\bar{z}}$ have aligned eigenspace V, we have

$$\begin{aligned}
V\Lambda_{\bar{z}} V^\top + \mathbb{V}_{z|\bar{z}} &= V\Lambda_z V^\top && (33)\\
\Longrightarrow \mathbb{V}_{z|\bar{z}} &= V\left(\Lambda_z - \Lambda_{\bar{z}}\right)V^\top, && (34)
\end{aligned}$$

which implies that the corresponding eigenvalues satisfy $\varepsilon_i = \lambda_i^z - \lambda_i^{\bar{z}}$, where $\bar{\lambda}_i, \varepsilon_i$ denote the i-th eigenvalues of $\mathbb{C}_{\bar{z}}, \mathbb{V}_{z|\bar{z}}$, respectively.

For the second part, we have

$$\begin{aligned}
\mathcal{L}(W) &= \mathbb{E}_{x,x^+}\frac{1}{2}\|Wz_x - \operatorname{sg}(z_{x^+})\|^2 && (35)\\
&= \mathbb{E}_{x,x^+}\frac{1}{2}\left(Wz_x - \operatorname{sg}(z_{x^+})\right)^\top\left(Wz_x - \operatorname{sg}(z_{x^+})\right) && (36)\\
&= \mathbb{E}_{x,x^+}\frac{1}{2}\left(z_x^\top W^\top W z_x - \operatorname{sg}(z_{x^+})^\top W z_x - z_x^\top W^\top \operatorname{sg}(z_{x^+}) + \operatorname{sg}(z_{x^+})^\top \operatorname{sg}(z_{x^+})\right) && (37)\\
&= \frac{1}{2}\mathbb{E}_{x,x^+}\left[\operatorname{Tr}\left(z_x^\top W^\top W z_x\right) - 2\operatorname{Tr}\left(\operatorname{sg}(z_{x^+})^\top W z_x\right) + \operatorname{Tr}\left(\operatorname{sg}(z_{x^+})^\top \operatorname{sg}(z_{x^+})\right)\right] && (38)\\
&= \frac{1}{2}\mathbb{E}_{x,x^+}\left[\operatorname{Tr}\left(W^\top W z_x z_x^\top\right) - 2\operatorname{Tr}\left(W z_x \operatorname{sg}(z_{x^+})^\top\right) + \operatorname{Tr}\left(\operatorname{sg}(z_{x^+})\operatorname{sg}(z_{x^+})^\top\right)\right] && (39)\\
&= \frac{1}{2}\left[\operatorname{Tr}\left(W^\top W \mathbb{E}_x z_x z_x^\top\right) - 2\operatorname{Tr}\left(W\mathbb{E}_{x,x^+} z_x z_{x^+}^\top\right) + \operatorname{Tr}\left(\mathbb{E}_{x^+} z_{x^+} z_{x^+}^\top\right)\right]. && (40)
\end{aligned}$$

Notice that $\mathbb{C}_z = \mathbb{E}_x z_x z_x^\top = \mathbb{E}_{x^+} z_{x^+} z_{x^+}^\top$ and

$$\begin{aligned}
\mathbb{E}_{x,x^+} z_x z_{x^+}^\top &= \mathbb{E}_{\bar{x}}\mathbb{E}_{x,x^+|\bar{x}} z_x z_{x^+}^\top && (41)\\
&= \mathbb{E}_{\bar{x}}\left(\mathbb{E}_{x|\bar{x}} z_x\right)\left(\mathbb{E}_{x|\bar{x}^+} z_{x^+}\right)^\top && (42)\\
&= \mathbb{E}_{\bar{x}} z_{\bar{x}} z_{\bar{x}}^\top && (43)\\
&= \mathbb{C}_{\bar{x}}. && (44)
\end{aligned}$$

Hence,

$$\mathcal{L}(W) = \frac{1}{2}\left[\operatorname{Tr}\left(W^\top W \mathbb{C}_z\right) - 2\operatorname{Tr}\left(W\mathbb{C}_{\bar{x}}\right) + \operatorname{Tr}\left(\mathbb{C}_z\right)\right]. \tag{45}$$

Taking partial derivative with respect to W, we get

$$\frac{\partial \mathcal{L}(W)}{\partial W} = W\mathbb{C}_z - \mathbb{C}_{\bar{x}}. \tag{46}$$

Let $\frac{\partial \mathcal{L}(W)}{\partial W} = 0$, we have $W^* = \mathbb{C}_{\bar{x}}\mathbb{C}_z^{-1}$. And $\mathbb{C}_{\bar{x}}\mathbb{C}_z^{-1} = V\Lambda_{\bar{z}}V^\top\left(V\Lambda_z V^\top\right)^{-1} = V\Lambda_{\bar{z}}\Lambda_z^{-1}V^\top = V\Omega V^\top$, where Ω is a diagonal matrix and $\Omega_{ii} = \frac{\bar{\lambda}_i}{\lambda_i}, i = 1, 2, \ldots, k$. □

B EXPERIMENTAL DETAILS

In this section, we provide the details and hyperparameters for SymSimSiam and variants of spectral filters.

B.1 DATASETS

We evaluate the performance of our methods on four benchmark datasets: CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), ImageNet-100 and ImageNet-1k (Deng et al., 2009). CIFAR-10 and CIFAR-100 are small-scale datasets, composed of 32×32 images with 10 and 100 classes, respectively. ImageNet-100 is a subset of ImageNet-1k containing 100 classes.

B.2 IMPLEMENTATION DETAILS

Unless specified otherwise, we follow the default settings in solo-learn (da Costa et al., 2022) on CIFAR-10, CIFAR-100 and ImageNet-100. For ImageNet, our implementation follows the official code of SimSiam (Chen & He, 2021), and we use the same settings. For a fair comparison, SimSiam with a learnable linear predictor is adopted as our baseline. With the original projector, SimSiam with a learnable linear predictor could not work, so we delete the last BN in the projector in this case. And we also list the results of SimSiam with the default predictor (we refer it as a learnable nonlinear predictor).

Data augmentations. The augmentation pipeline is RandomResizedCrop with with scale in (0.2, 1.0), RandomHorizontalFlip with probability 0.5, ColorJitter (brightness (0.4), contrast (0.4), saturation (0.4), hue (0.1)) with probability 0.8 and RandomGray with probability 0.2. For ImageNet-100 and ImageNet-1k, Gaussian blurring (Chen et al., 2020) with an applying probability 0.5 is also used.

Optimization. SGD is used as the optimizer with momentum 0.9 and weight decay 1.0×10^{-5} (1.0×10^{-4} for ImageNet-1k). The learning rate adopts the linear scaling rule (lr×BatchSize/256) with the base learning rate of 0.5 (0.05 for ImageNet-1k). After 10 epochs warmup training, we use the cosine learning rate decay (Loshchilov & Hutter, 2017). We use batch size 256 on CIAFR-10 and CIFAR-100; 128 on ImageNet-100 and 512 on ImageNet-1k.

Linear evaluation. For the linear evaluation, we evaluate the pre-trained backbone network by training a linear classifier on the frozen representation. For CIFAR-10, CIFAR-100 and ImageNet-100, the linear classifier is trained using SGD optimizer with momentum = 0.9, batch size = 256 and initial learning rate = 30.0 for 100 epochs. The learning rate is divided by 10 at epochs 60 and 80. For ImageNet-1k, following the official code of SimSiam (Chen & He, 2021), we train the linear classifier for 90 epochs with a LARS optimizer (You et al., 2017) with momentum = 0.9, batch size = 4096, weight decay = 0, initial learning rate = 0.1 and cosine decay of the learning rate.

B.3 EXPERIMENTAL SETTING FOR FIGURES

For the implementations of BYOL, SimSiam, SwAV and DINO on CIFAR-10 and CIFAR-100, we use codes and all default settings in the open-source library solo-learn (da Costa et al., 2022). For SwAV, we do not store features from the previous batches to augment batch features (queue_size = 0) for the consistency of training loss. We adopt ResNet-18 as the backbone encoder. The dimensions of the outputs of BYOL, SimSiam, SwAV and DINO are 256, 2048, 3000 and 4092, respectively.

In Figure 3, we use the alignment metric to measure the eigenspace alignment of the online and target outputs, i.e., $\mathbb{C}_p$ and $\mathbb{C}_z$. The intuition is that, given u_i as an eigenvector of $\mathbb{C}_z$, if it is also an eigenvector of $\mathbb{C}_p$, then $\mathbb{C}_p u_i = \lambda_i' u_i$ is in the same direction as u_i. Thus, the higher the cosine similarity between u_i and $\mathbb{C}_p u_i$, the more aligned between the eigenspace of $\mathbb{C}_p$ and $\mathbb{C}_z$. Formally, we define the alignment between the eigenspace of $\mathbb{C}_p$ and $\mathbb{C}_z$ as

$$Alignment(\mathbb{C}_p, \mathbb{C}_z) = \frac{1}{m} \sum_{i=1}^{m} \frac{u_i^T}{\|u_i\|_2} \frac{\mathbb{C}_p u_i}{\|\mathbb{C}_p u_i\|_2}, \tag{47}$$

where u_i is eigenvector corresponding to the i-th largest eigenvalue λ_i^z of $\mathbb{C}_z$, $i = 1, 2, \ldots, m$. And m is set to 512 for SimSiam. SwAV and DINO and 128 for BYOL. The sum of the first m eigenvalues is greater than 99.99% of the sum of all eigenvalues. Therefore, we can think that the space span by the first m eigenvectors is a good approximation to the original eigenspace. And in Figures 4(b) & 8(b), the first 512 point pairs are displayed (256 for BYOL). In Figure 5(a) & 5(b) the first 128 and 512 point pairs are shown for BYOL and SimSiam, respectively.

B.4 PSEUDOCODE

Here, we provide the pseudocode for SymSimSiam (Algorithm 1) and variants of spectral filters (Algorithm 2).

Algorithm 1 SymSimSiam: Pseudocode in a PyTorch-like style.

```
# f: backbone + projection mlp
# c: center (1-by-k)
# m: momentum 0.9

for x in loader:                # load a minibatch x with n samples
    x1, x2 = aug(x), aug(x)     # two random augmentations
    z1, z2 = f(x1), f(x2)       # projections, n-by-k
    z1 = normalize(z1, dim=1)   # l2-normalize
    z2 = normalize(z2, dim=1)   # l2-normalize

    update_center(cat(z1,z2))
    z1, z2 = z1-c, z2-c

    loss = -(z1*z2).sum(dim=1).mean()

    loss.backward()
    update(f.params)            # SGD update

@torch.no_grad()
def update_center(z):
    c = m * c + (1-m) * z.mean(dim=0)
```

Algorithm 2 Variants of spectral filters: Pseudocode in a PyTorch-like style.

```
# f: backbone + projection mlp
# g: spectral filter
# location: the location of spectral filter (online or target)

for x in loader:                # load a minibatch x with n samples
    x1, x2 = aug(x), aug(x)     # two random augmentations
    z1, z2 = f(x1), f(x2)       # projections, n-by-k

    loss = L(z1,z2,location)/2 + L(z2,z1,location)/2

    loss.backward()
    update(f.params)            # SGD update

def L(p,z,location):            # loss function
    if location == "online":    # online predictor
        with torch.no_grad():
            u, s, vh = svd(p)
            w = vh.T @ diag(g(s)) @ vh
        p = p @ w.detach()
    else:                       # target predictor
        with torch.no_grad():
            u, s, vh = svd(z)
            z = u @ diag(g1(s)) @ vh  # g1(t) = tg(t)

    z = z.detach()              # stop gradient
    p = normalize(p, dim=1)     # l2-normalize
    z = normalize(z, dim=1)     # l2-normalize
    return -(p*z).sum(dim=1).mean()
```

Table 4: Linear probing accuracy (%) of BYOL with different target predictors on CIFAR-10/100.

Predictor	CIFAR10			CIFAR100		
	100ep	200ep	400ep	100ep	200ep	400ep
Learnable (nonlinear)	82.14	87.89	91.24	53.92	60.75	67.41
Learnable (linear)	80.81	86.58	89.73	49.65	57.53	60.66
$h(\sigma) = \sigma^{-0.3}$	**85.73**	**89.26**	**90.73**	57.56	61.89	**65.88**
$h(\sigma) = \sigma^{-0.5}$	85.46	89.24	90.30	58.12	62.51	65.16
$h(\sigma) = \sigma^{-0.7}$	85.08	88.21	90.01	58.20	**62.58**	65.72
$h(\sigma) = \sigma^{-0.1}$	84.64	88.29	89.96	**58.40**	62.06	65.04

Table 5: Training time and memory comparison of different methods on CIFAR-10.

Method	Total time for 20 epochs	GPU memory
SimSiam	527 s	3043 MiB
$g(\sigma) = \log(1+\sigma)$	532 s	3069 MiB
$h(\sigma) = \sigma^{-0.3}$	527 s	3069 MiB

C Additional Results Based on BYOL Framework

In the main paper, we conduct experiments based on SimSiam framework. Here, we also gather some results of variants of target high-pass filters based on BYOL framework. We adopt ResNet-18 as the backbone encoder and use the default setting in da Costa et al. (2022).

Table 4 shows that our target high-pass filters can often outperform BYOL with a large margin at earlier epochs. In particular, on CIFAR-10, the target filter $h(\sigma) = \sigma^{-0.3}$ improves BYOL with learnable linear predictor by 4.92%, 2.68% and 1.00% with 100, 200 and 400 epochs, respectively. On CIFAR-100, the filter $h(\sigma) = \sigma^{-1}$ is better 8.75% than the baseline with 100 epochs, the filter $h(\sigma) = \sigma^{-0.7}$ improves the baseline by 5.05% with 200 epochs and $h(\sigma) = \sigma^{-0.3}$ improves the baseline by 5.22% with 400 epochs.

D Cost Comparison

We compare the training speeds and GPU memory usages of different methods (SimSiam, the online filter $g(\sigma) = \log(1+\sigma)$ and the target filter $h(\sigma) = \sigma^{-0.3}$) on CIFAR-10. We perform our experiments on a single RTX 2080ti GPU.

As shown in Table 5, compared to the original SimSiam, the GPU memory usage of our methods only increase a little (26 MiB). SimSiam and the target filter $h(\sigma) = \sigma^{-0.3}$) have the same training time for 20 epochs.

E Additional Figure

In Figure 9, we compare the training loss between SimSiam and SymSimSiam on CIFAR-10 and CIFAR-100. We can see that the loss of SimSiam is always larger than SymSimsiam and does not consistently decrease. Instead, the alignment loss of SymSimSiam continues to decrease.

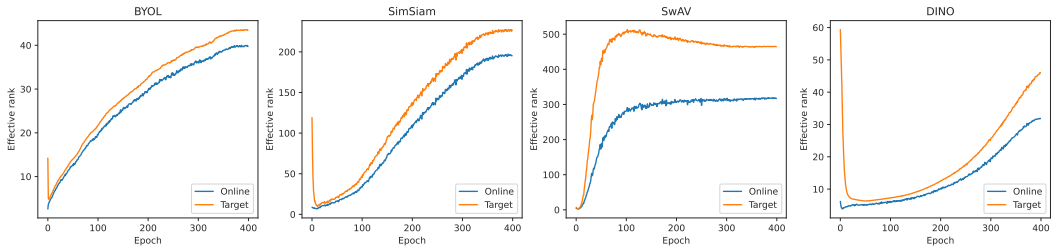
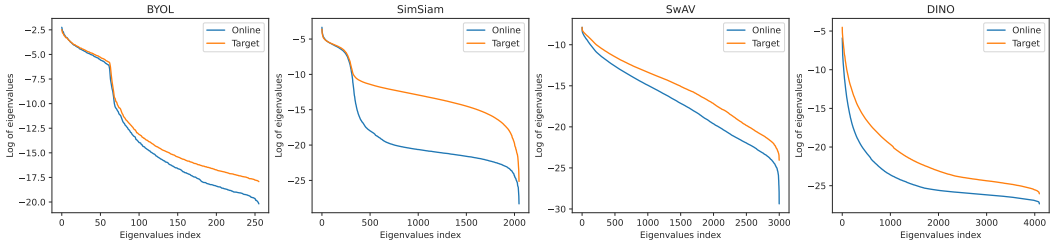
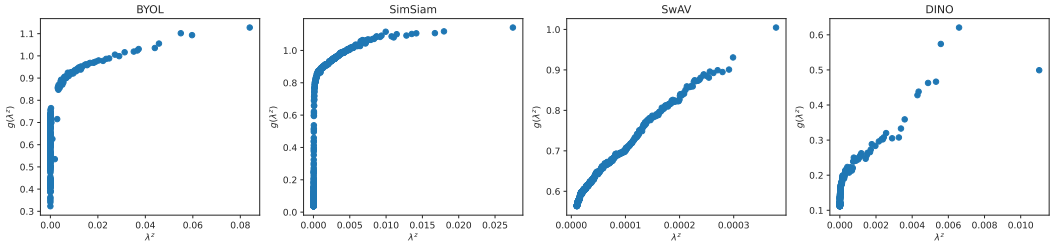


Figure 7: The effective rank of the normalized outputs of the online and target branches along the training dynamics on CIFAR-100.

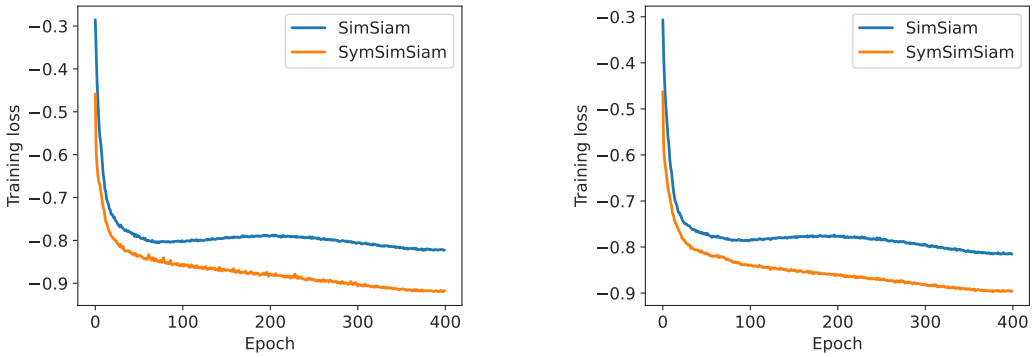


(a) Eigenvalues of the feature correction matrices of the online and target outputs.



(b) The corresponding online spectral filter function $g(\lambda_i^z) = \sqrt{\lambda_i^p / \lambda_i^z}$.

Figure 8: Rank difference experiments and spectral filters on CIFAR-100. (a): Comparison of the eigenvalues computed from two branch outputs. There is a clear difference in eigenvalues, and the eigenvalue distribution of the online branch is more biased towards larger values. (b): The spectral filter $g(\lambda_i^z) = \sqrt{\lambda_i^p / \lambda_i^z}$, where λ_i^p, λ_i^z are eigenvalues of online and target correlation matrices $\mathbb{C}_p, \mathbb{C}_z$, respectively. The spectral filters of all methods are nearly monotonically increasing *w.r.t.* λ_i^z.



(a) CIFAR-10

(b) CIFAR-100

Figure 9: Training loss of SimSiam and SymSimSiam on CIFAR-10 and CIFAR-100.