Understanding and Improving the Role of Projection Head in Self-Supervised Learning

Kartik Gupta[†][*] **Thalaiyasingam Ajanthan**[†‡], **Anton van den Hengel**[‡], **Stephen Gould**[†‡]
[†] Australian National University, [‡] Amazon
[†]kartik.gupta@anu.edu.au

Abstract

Self-supervised learning (SSL) aims to produce useful feature representations without access to any human-labeled data annotations. Due to the success of recent SSL methods based on contrastive learning, such as SimCLR, this problem has gained popularity. Most current contrastive learning approaches append a parametrized projection head to the end of some backbone network to optimize the InfoNCE objective and then discard the learned projection head after training. This raises a fundamental question: Why is a learnable projection head required if we are to discard it after training? In this work, we first perform a systematic study on the behavior of SSL training focusing on the role of the projection head layers. By formulating the projection head as a parametric component for the InfoNCE objective rather than a part of the network, we present an alternative optimization scheme for training contrastive learning based SSL frameworks. Our experimental study on multiple image classification datasets demonstrates the effectiveness of the proposed approach over alternatives in the SSL literature.

1 Introduction

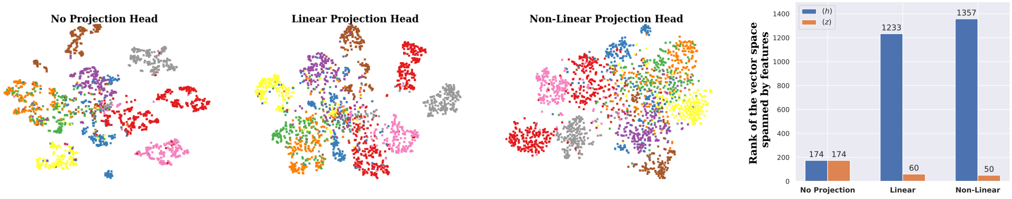
The ultimate goal of self-supervised learning (SSL) is to obtain generalizable features from the information inherent to massive amounts of unlabelled data in a task-agnostic manner. These features can then be used to perform various downstream tasks using only a minimal amount of supervised training and a small set of task-specific label data. The SSL task is typically formulated as contrastive learning, where the idea is to learn features that remove the effect of data augmentations applied to the input data. Here, the intuition is that data augmentations cover the style space, which is often irrelevant to the downstream tasks. As one can imagine, prior knowledge of the downstream tasks is necessary to design meaningful data augmentations, and even then, it is a challenging problem.

Nevertheless, contrastive SSL methods are successful in many applications [3, 5, 24]. An important architectural choice in the majority of these methods is the use of a multi-layer perceptron (MLP) appended to the network (i.e., projection head) to project the backbone features into a low dimensional space before applying the contrastive loss. This projection head is discarded after training as the projected features have been found to be inferior in terms of generalization performance and the backbone features are directly used for the downstream tasks [2]. Despite its practical importance, the role of the projection head in SSL methods is not well-understood.

In this work, we attempt to empirically understand why a learnable projection head is required if we are to discard it after training? We would like to highlight two important observations: First, the projection head is a low-rank mapping. Second, the null space of the projection head is useful for generalization. Based on these observations, we hypothesize that *the projection head implicitly learns to choose a subspace of features to apply the contrastive loss*. This subspace selection addresses the shortcomings of the contrastive loss (*e.g.*, sub-optimal data augmentations), however, this property

[*] Work done partially during internship at Amazon, Adelaide.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) (b) (c) (d)

Figure 1: *t-SNE plots for SimCLR trained models on CIFAR-10 using ResNet-50 with different projection head configurations in (a), (b) and (c). (d) Rank of projection head output space ($\mathcal{Z}$) and backbone encoder output space ($\mathcal{H}$), obtained using PyTorch (`matrix_rank`).*

emerges as a side-effect. In particular, the implicit subspace selection enables the projection head output to minimize the contrastive loss, while allowing backbone to learn generalizable features.

Based on this hypothesis, we argue that the data-dependent subspace selection should be considered as part of the SSL loss function and this behavior should be enforced rather than relied upon. To this end, we formulate self-supervised learning as a bilevel optimization problem. Here, at each training step, the inner-optimization selects the best subspace for the contrastive loss (by optimizing the projection head) and the outer-optimization performs gradient descent on the backbone network.

We perform several experiments on CIFAR-10, STL-10, TinyImageNet, and ImageNet datasets with SimCLR [2] and SimSiam [6] to understand the role of the projection head. Later, we evaluate our modified optimization scheme for SSL on CIFAR-10, CIFAR-100 and TinyImageNet datasets with the SimCLR method. Our results, obtained by viewing the projection head as part of the loss, show better generalization over SimCLR and validates the hypothesis on the role of the projection head.

2 Preliminaries and Notations

Given a dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$ and data augmentations $\mathcal{T}$, the SSL learning problem can be written as:

$$\min_{f,g} L(g \circ f; \mathcal{D}, \mathcal{T}) \triangleq \mathop{\mathbb{E}}_{\substack{\mathbf{x}, \mathcal{B}_{\mathbf{x}} \sim \mathcal{D} \\ t_1, t_2 \sim \mathcal{T}}} \left[\ell(\mathbf{z}^1, \mathbf{z}^2; \mathcal{Z}_{\mathbf{x}}) \right] , \tag{1}$$

where $\mathbf{z}^j = g \circ f(t_j(\mathbf{x}))$ for $j \in \{1, 2\}$, $\mathcal{Z}_{\mathbf{x}} = \{g \circ f(t_j(\mathbf{y})) \mid \mathbf{y} \in \mathcal{B}_{\mathbf{x}}, j \in \{1, 2\}\}$. Here, g denotes the projection head, f is the (feature) encoder, $\mathcal{B}_{\mathbf{x}}$ is a mini-batch sampled from $\mathcal{D}$ that does not include $\mathbf{x}$, and ℓ is the example-wise contrastive loss function. While there are various contrastive loss functions exist [23], InfoNCE [2] is the most popular. The contrastive loss encourages the embeddings of two augmented views of the same training example to be closer while pushing embeddings of other training examples apart. In practice, f and g in Eq. (1) are parametrized using neural networks and $\min_{g,f}$ denotes minimizing the parameters of the respective neural networks.

Let us denote the backbone features by $\mathbf{h} = f(\mathbf{x}) \in \mathbb{R}^m$ and the projection head output as $\mathbf{z} = g(\mathbf{h}) \in \mathbb{R}^d$, where $m \geq d$ in practice. Moreover, let $\mathcal{H} \subseteq \mathbb{R}^m$ and $\mathcal{Z} \subseteq \mathbb{R}^d$ be the vector spaces spanned by the backbone features and projection head outputs, respectively. Once trained, the projection head g is stripped away [2] and the backbone feature $\mathbf{h}$ is used for the downstream tasks.

3 A Closer Look at the Role of the Projection Head

We start by conducting an empirical study on various choices of projection heads available for contrastive learning on multiple datasets such as CIFAR-10, STL-10, TinyImageNet and ImageNet using SimCLR [2] and SimSiam [7] method. We provide experimental details for the study in Section C in Appendix. As shown in [2], non-linear projection head consistently yields better performance than linear projection head and identity projection head. Specifically, this raises a non-trivial question how does the projection head aid the training of SSL objective. To this end, we first analyse the rank of the learned feature spaces obtained via different projection heads.

Feature Representation with Different Projection Heads. To further analyse the representation quality of backbone features learned with various projection head configurations, we present t-SNE [27] plots in Fig. 1 with different projection heads using SimCLR training on the CIFAR-10 dataset. Although, the feature representations learned directly on the backbone features (no projection head) are more tightly clustered, the issue of "feature collapse" is easy to observe. Whereas a non-linear projection head yields features spread across the whole space with less evidence of "feature collapse".

In order to validate this hypothesis, we further provide the rank for the output of the projection head and the network backbone features in Fig. 1 (d). This clearly shows that while all projection head variants yield low-rank backbone features, the rank of $\mathcal{H}$ for the no projection head case is lower than the non-linear projection head configuration. Another consistent observation reveals that the rank of projected features ($\mathcal{Z}$) is consistently lower than that of the backbone features ($\mathcal{H}$). This indicates that the contrastive learning loss tends to result in low-rank outputs $\mathcal{Z}$.

Furthermore, there is a clear correlation between the quantity $\mathrm{rank}(\mathcal{H}) - \mathrm{rank}(\mathcal{Z})$ (which we call *rank deficit*) and the generalization performance. Specifically, the rank deficit from $\mathcal{H}$ to $\mathcal{Z}$ increases from no projection head (0), to linear (1173), to the non-linear projection head (1307) (refer to Fig. 1 (d)). The same order is apparent in the linear evaluation performance in [6]. We further analyse the generalization performance of the null space of the projection head to understand the additional information in $\mathcal{H}$ ignored by the projection head.

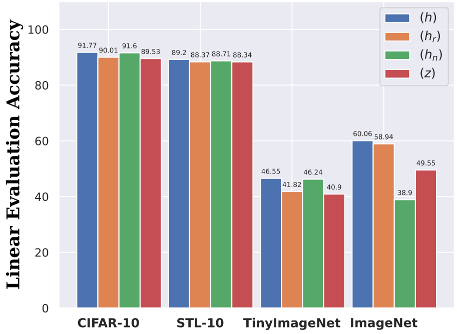


Figure 2: *Linear evaluation accuracy on different datasets using various feature components.*

Null Space Analysis for Linear Projection. For simplicity, we consider the linear projection head without the bias term and analyse the generalization performance of its null space (*i.e.*, the subspace of $\mathcal{H}$ which is completely ignored by g). Let $\boldsymbol{A} \in \mathbb{R}^{d\times m}$ be the weight matrix of the linear projection head g, where m is the dimension of the backbone features ($\mathbf{h}$) and d is projection head output dimension ($\mathbf{z}$). Therefore $\mathbf{z} = \boldsymbol{A}\mathbf{h}$ and we intend to understand how $\boldsymbol{A}$ decomposes the backbone feature space. To this end, any vector $\mathbf{h} \in \mathbb{R}^m$ can be written as a sum of two orthogonal components $\mathbf{h} = \mathbf{h}_r + \mathbf{h}_n$ such that $\mathbf{h}_r \in \mathcal{R}(\mathbf{A}^T)$ and $\mathbf{h}_n \in \mathcal{N}(\mathbf{A})$, where $\mathcal{R}(\mathbf{A})$ and $\mathcal{N}(\mathbf{A})$ are the range (*i.e.*, column space) and the null space of $\mathbf{A}$, respectively. Precisely, $\mathbf{h}_r = \boldsymbol{A}^+\boldsymbol{A}\mathbf{h}$ and $\mathbf{h}_n = \mathbf{h} - \mathbf{h}_r$, where $\boldsymbol{A}^+$ is the right inverse, *i.e.*, $\boldsymbol{A}^+ = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}$. Note that any vector in the null space maps to $\mathbf{0}$ and therefore would be ignored when using $\mathbf{z}$ for the downstream tasks. To analyse the null space, we use the above decomposition to obtain the component corresponding to the null space for each feature vector $\mathbf{h}$. We then evaluate the performance of different components $\mathbf{h}_r$, $\mathbf{h}_n$, $\mathbf{h}$ and $\mathbf{z}$ over pre-trained SSL models on different datasets (SimSiam for ImageNet and SimCLR for all other datasets) in Fig. 2. While the full backbone feature $\mathbf{h}$ is the best performing, the null space $\mathbf{h}_n$ is competitive and except in ImageNet, it outperforms the projection head output $\mathbf{z}$. Even in ImageNet, the null space is significantly better than the random classification. This clearly shows that the null space has useful information for generalization without any direct influence on the contrastive loss.

From the above analysis, we want to highlight the following observations: (1) The null space of the projection head is generalizable and sometimes it performs better than the projection head output $\mathbf{z}$. Refer to Fig. 2. (2) There is a clear positive correlation between the rank deficit from $\mathcal{H}$ to $\mathcal{Z}$ (*i.e.*, $\mathrm{rank}(\mathcal{H}) - \mathrm{rank}(\mathcal{Z})$) and the generalization performance of the backbone features. We hypothesize that *the learnable projection head is a way of mitigating the shortcomings of the contrastive loss*. Specifically, the projection head implicitly learns to select a subspace of the features $\mathcal{H}$ and (non)-linearly map them to $\mathcal{Z}$ to apply the contrastive loss. In this way, the contrastive loss is minimized on $\mathcal{Z}$, which is encouraged to become style-invariant (hence, sensitive to the sub-optimal choice of data augmentations), whereas the backbone features $\mathcal{H}$ are not forced to be style-invariant and as a result can generalize better. In this work, we intend to make use of the above interpretation regarding the role of projection head to improve the SSL framework.

4 Self-Supervised Learning with Adaptive Contrastive Loss

As discussed in the previous section, if the projection head g chooses the best subspace to apply the contrastive loss and is also stripped away after training, then we believe that g should be considered as a part of the loss function rather than a part of the network. Specifically, we argue that the SSL objective should be treated as the best contrastive loss that can be obtained by searching over all (fixed-dimensional) subspaces of the features. As such we define the new objective for SSL as $L^\star(f;\mathcal{D},\mathcal{T}) \triangleq \min_g L(g\circ f;\mathcal{D},\mathcal{T})$, allowing us to re-write the SSL objective in Eq. (1) as:

$$\min_f L^\star(f;\mathcal{D},\mathcal{T}) = \min_f \min_g L(g\circ f;\mathcal{D},\mathcal{T}) \ . \tag{2}$$

Note that if we have access to the minimization oracle of L, then the optimization problems described by Eq. (1) and Eq. (2) are equivalent. However, such an oracle does not exist in practice. As such, we propose an iterative optimization algorithm for the above formulation.

Iterative Optimization. Notice that in Eq. (2) we have separated the optimization with respect to g and f, and the iterative updates can be written as:

$$\begin{aligned} g^{k+1} &= \operatorname*{argmin}_{g} L(g \circ f^{k}; \mathcal{D}, \mathcal{T}) \;, \\ f^{k+1} &= f^{k} - \eta \nabla_{f} L(g^{k+1} \circ f^{k}; \mathcal{D}, \mathcal{T}) \;, \end{aligned} \tag{3}$$

where k denotes the iteration number, $\eta > 0$ is the learning rate, and $\nabla_f L$ denotes the gradient of L with respect to the parameters of f. This bilevel optimization is computationally expensive as the inner-optimization (wrt. g) needs to be solved for every iteration of the outer optimization (wrt. f).

Therefore, for computational efficiency, we resort to a stochastic optimization strategy where for each iteration k a mini-batch of data $\mathcal{D}^k \subset \mathcal{D}$ is used instead of the whole dataset. However, this could lead to g quickly overfitting to the small mini-batch, leading to sub-optimal results. To alleviate such overfitting and allow stability, we resort to a truncated optimization with a proximal term by fixing the number of iterations to the inner optimization and use momentum based gradients. Such an approximation is common in deep learning (*e.g.*, [11]). Our final iterative updates can be written as:

$$\begin{aligned} g^{k+1} &\approx \operatorname*{argmin}_{g} L(g \circ f^{k}; \mathcal{D}^{k}, \mathcal{T}) + \lambda \left\| g - g^{k} \right\|_2^2 && l \text{ steps of SGD,} \\ f^{k+1} &= f^{k} - \eta \nabla_{f} L(g^{k+1} \circ f^{k}; \mathcal{D}^{k}, \mathcal{T}) && 1 \text{ step SGD,} \end{aligned} \tag{4}$$

where any standard stochastic gradient descent (SGD) algorithm can be used. Here, $\lambda > 0$ is the strength of the proximal term and $\|\cdot\|_2^2$ denotes the L_2 norm.

Note that the optimization with respect to g can be performed quickly as it is a small MLP, and run for a small number of iterations (typically, $l \leq 10$ in our experiments). Then, the updated g^{k+1} is used to compute the gradient with respect to the backbone network parameters. Overall, the computational complexity of one iteration of our algorithm is the same as the standard back-propagation.

Practical Benefits. According to Eq. (4), it is clear that at every iteration of our algorithm, we first update the loss parameters g (via inner-optimization) and use the updated loss to perform gradient descent on f. In this way, the most recent update on g (*i.e.*, g^{k+1}) is immediately propagated to f. Therefore, based on our hypothesis in Sec. 3, our approach modifies the loss to improve its generalizability (*i.e.*, selects the best subspace to apply the contrastive loss for each mini-batch) and takes a gradient step on the updated loss.

5 Comparisons against Baselines

As a proof of concept, we provide the experimental evaluations of our proposed optimization scheme against the standard training regime of SimCLR framework on CIFAR-10/100, and TinyImageNet datasets in Table 1. As explained above, we use both linear evaluation and KNN based evaluation to show these comparisons. Our approach achieves better generalization performance using both these evaluations consistently on CIFAR-10/100 and TinyImageNet datasets. Infact our method reaches near optimum linear evaluation accuracy at just 500 epochs, which the standard SimCLR training procedure can achieve after nearly double the training cost. This empirically demonstrates the capability of faster convergence of our alternative optimization scheme. The performance gain of our method over the SimCLR trained using non-linear projection head is especially significant ($\approx 2\%$ on CIFAR100 and TinyImageNet dataset) in case of KNN evaluation.

Methods	**CIFAR-10**		**CIFAR-100**		**TinyImageNet**	
	KNN	Linear	KNN	Linear	KNN	Linear
No Projection	85.69	88.07	47.67	54.78	28.20	35.04
SimCLR [2]	87.43	91.11	56.10	68.01	38.44	50.64
Random Proj.	86.57	90.75	51.81	63.25	32.88	46.40
DirectCLR [18]	86.93	90.19	52.23	65.20	33.08	46.71
Ours	**88.53**	**91.97**	**58.08**	**69.12**	**40.48**	**51.72**

Table 1: *Comparisons against SimCLR on CIFAR-10/100 and TinyImageNet datasets using ResNet-50 architecture.*

We also compare our method against alternative mechanism to a standard approach of performing contrastive learning using non-linear projection head in Table 1. Our method consistently performs better than recently proposed method namely DirectCLR [18] as well alternative approach of performing SimCLR with a fixed randomly initialized projection head.

References

[1] 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.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[3] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big Self-Supervised Models are Strong Semi-Supervised Learners. 2020.

[4] Ting Chen, Calvin Luo, and Lala Li. Intriguing properties of contrastive losses. *arXiv preprint arXiv:2011.02803*, 2020.

[5] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[6] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. 2021.

[7] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[8] Ching-Yao Chuang, Joshua Robinson, Lin Yen-Chen, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *arXiv preprint arXiv:2007.00224*, 2020.

[9] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. *arXiv preprint arXiv:2104.14548*, 2021.

[10] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for Self-Supervised Representation Learning. *arXiv preprint arXiv:2007.06346*, 2020.

[11] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[12] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

[14] Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *arXiv preprint arXiv:2106.04156*, 2021.

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

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

[17] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2021.

[18] Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. *arXiv preprint arXiv:2110.09348*, 2021.

[19] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *arXiv preprint arXiv:1902.06162*, 2019.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

[21] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. 2015.

[22] Junnan Li, Pan Zhou, Caiming Xiong, Richard Socher, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020.

[23] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[24] Pierre H. Richemond, Jean-Bastien Grill, Florent Altché, Corentin Tallec, Florian Strub, Andrew Brock, Samuel Smith, Soham De, Razvan Pascanu, Bilal Piot, and Michal Valko. BYOL works even without batch statistics. *arXiv preprint arXiv:2010.10241*, 2020.

[25] Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised Learning Dynamics without Contrastive Pairs. *arXiv:2102.06810 [cs]*, February 2021. arXiv: 2102.06810.

[26] Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Self-supervised learning from a multi-view perspective. *arXiv preprint arXiv:2006.05576*, 2020.

[27] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[28] Julius von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. *arXiv preprint arXiv:2106.04619*, 2021.

[29] Tongzhou Wang and Phillip Isola. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. *arXiv preprint arXiv:2005.10242*, 2020.

[30] Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. Theoretical analysis of self-training with deep networks on unlabeled data. *arXiv preprint arXiv:2010.03622*, 2020.

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

Appendix

A Related Work

Self-supervised learning (SSL) literature has become increasingly popular in the last few years with the promise of boosting performance in various application domains where obtaining large volumes of unlabeled data is cheap, including vision and language. Due to space constraints, we briefly discuss previous works that are closely related to our work and/or inspired our thinking, and we refer the interested reader to the surveys [19, 17, 23] for a comprehensive study.

Here, we mainly consider example-wise contrastive learning approaches, which can be categorized into methods that require explicit negative samples [2, 3, 15, 5, 10, 29] and those that are negative-sample-free [13, 7, 31]. In the former methods, for a given input sample, all other samples in the mini-batch are regarded as negatives and the loss is encouraged to pull various augmentations of the same sample together in the feature space while pushing the features from the other samples (*i.e.*, the negatives) apart. This has two main issues. First, the notion of negative samples is unclear without the label information; and, second, these methods require large mini-batches to compute effective statistics for training.

Different strategies have been explored to obtain better positive and negative samples to apply the contrastive loss within the SSL framework [8, 9] On the other hand, the latter methods circumvent the requirement of negative samples altogether by introducing asymmetry in the network architecture [13, 7] or by modifying the loss function [31]. In both types of methods, the projection head is an integral part of the architecture design and it is discarded after training. This is true even for SSL approaches that do not use example-wise contrastive learning [1, 22].

There are some efforts to theoretically understand the SSL methods [30, 14], the role of data augmentation [28, 26], and some empirical analyses of the contrastive loss [4] and the predictor in the so-called BYOL framework [25]. Nevertheless, to the best of our knowledge, there is no work that attempts to understand the role of the projection head in the learning process, or indeed, why generalizability improves when it is stripped away for downstream tasks.

Closest to our work is the concurrent work DirectCLR [18], which shows that the projection head becomes low-rank due to strong data augmentations that distort the content information in the input data. Nevertheless, their final approach is to directly optimize the backbone features with a fixed projection head. Whereas we empirically show a trainable projection head enhances performance and introduce an improved optimization scheme for SSL. Even though DirectCLR is a concurrent work, we compare it in our experiments and demonstrate that while a fixed projection yields better generalization in the early stages of training, a learnable projection head eventually outperforms it.

B Limitations

This work is the first of a kind study on the role of the projection head in the SSL training and thus it has certain limitations and open questions for future research. Although contrastive learning-based SSL has made significant progress in the pre-training domain, the underlying procedure remains more so as a black-box system. Some recent attempts [30, 14] have been made in this direction but we believe there is still a gap between the theoretical understanding of contrastive learning-based SSL and its empirical success.

Our work mainly attempts to understand the behavior of SSL training by means of extensive empirical analysis, though the theoretical understanding of the projection head remains part of the aforementioned future work. Even though our proposed alternative optimization scheme has been shown to yield better generalization capabilities, we do not provide any theoretical guarantees for the success of our method in this paper. The proposed method aims to improve the trainability of the projection head and in turn SSL training procedure.

Our observations are consistent on multiple small-scale or large-scale datasets but the alternative optimization scheme has been considered only on small-scale datasets namely CIFAR-10, CIFAR-100 and TinyImageNet. Nevertheless, the observations and analysis that we presented in this paper provide a foundation for further work, and we believe they are valuable to other researchers in the field.

C Experimental Details

Datasets and Networks. In order to showcase the better trainability via the proposed method, we perform experiments on image classification datasets such as CIFAR-10, composed of 32×32 images with 10 and 100 classes, respectively, and TinyImageNet [21], a reduced version of ImageNet, composed of 200 classes with images resized to size 64×64, consisting of 100K training images and 10K testing image. We use ResNet-50 [16] network architecture to evaluate the SSL frameworks.

Training and Evaluation Hyperparameters. For all datasets, the output dimension for projection head $d = 128$, except ImageNet where $d = 2048$. We use the ResNet-50 backbone with $m = 2048$ and experiment with different projection heads. We describe the different projection head configurations used for these experiments below.

- *No Projection Head*: The projection head is simply removed from the original training framework and the loss is directly optimized on the backbone features $\mathbf{h}$.
- *Linear Projection Head*: A linear layer with bias term is used for projecting backbone features into a lower-dimensional subspace.
- *Non-Linear Projection Head*: A 2-layer MLP with ReLU non-linearity and batch normalization is used. The hidden layer dimension is 512 for all the datasets except ImageNet where it is 2048.

For both the datasets, we use the Adam optimizer [20]. We compared our method with arguably the most popular current SSL method, SimCLR [2]. For our proposed optimization scheme, we use the same learning rate with five inner optimization steps. All the experiments are performed using

InfoNCE loss with a temperature scale value of 0.5. We use 500 epochs with learning rate 10^{-3} and weight decay 10^{-6} for both CIFAR-10 and TinyImageNet. We use a mini-batch size of 256 images for CIFAR-10 dataset and 512 images for TinyImageNet dataset. All the experimental comparisons are performed with non-linear projection head or linear projection head. The dimension of the hidden layer of the non-linear projection head g is 512. The output of the embedding size is 128 for both CIFAR-10/100 and TinyImageNet dataset. All the experiments are performed using NVIDIA Tesla V100 GPUs.

For the purpose of evaluating the generalization capabilities of pre-trained features learned using SSL, we employ KNN based evaluation with 200 neighbours and linear evaluation where a linear layer is trained on pre-trained backbone features. For linear evaluation, we train the linear layer for 200 epochs using Adam optimizer with learning rate 10^{-3} and weight decay 10^{-6} for both CIFAR-10/100 and TinyImageNet.

Augmentations used. Similar to SimCLR, to generate the augmented pairs we extract crops with a random size from 0.2 to 1.0 of the original area and a random aspect ratio from 0.75 to 1.33 of the original aspect ratio for TinyImageNet. For CIFAR-10/100, we randomly extract the crops of size 32×32. We apply grayscaling to the samples with the probability 0.2 for CIFAR-10/100 dataset and 0.1 for TinyImageNet dataset. We use color jittering with brightness, contrast, saturation, and hue configuration of $(0.4, 0.4, 0.4, 0.1)$ with probability 0.8. We also apply horizontal flipping to the image pairs with 0.5 probability.

D Analysis on Different Procedures to Obtain the Projection Head

We now provide an experimental comparison on different possible fixed projection heads that can be used to replace a trainable projection head to further validate that a trainable projection head indeed aids in better training of SimCLR training procedure and similar benefits cannot be achieved with a fixed projection head. In this study, we resort to KNN evaluation for comparative analysis and linear projection heads and employ ResNet-50 architecture. Our analysis on different projection heads can be divided into *fixed projection head* and *moving projection head*. We now discuss the multiple variants of different projection head schemes that we use for this study.

Fixed Projection Head. In this setting, we keep the projection head parameters fixed throughout the training procedure of SimCLR. The fixed projection head can be obtained in the following variations:

- *Random Initialization:* A linear projection head obtained via standard network initialization [12] is employed in this case.
- *SimCLR based pre-training:* Here, we evaluate a fixed projection head obtained at the end of standard SimCLR training. Note, the idea here is to evaluate if there exists an optimal projection head that can be used in the training procedure of SimCLR as a fixed projection head (but possibly suboptimal with respect to the current backbone network parameters).
- *DirectCLR:* This is a recent concurrent work [18] where the proposed fixed projection head takes the form of a fixed low-rank diagonal matrix and is shown to outperform a trainable linear projection head. Their proposed fixed projection head essentially translates into SimCLR objective onto a subset of backbone features $\mathbf{h}$.

Moving Projection Head. In the standard training procedure of SimCLR framework, projection head parameters are updated in the same backward pass as the backbone encoder. Although, as shown via our proposed approach that this objective can achieve more efficiently via an alternating optimization scheme. We now present some other alternatives for the training of the projection head which are described below.

- *PCA based Projection:* Since our empirical observation demonstrates the projection head is low-rank. It is straightforward to estimate the projection head using principal components of $\mathcal{H}$. Therefore, to evaluate the efficacy of principal components as projection head, we perform PCA at the end of each epoch on a subset of the training dataset and use top-k or bottom-k principal components as fixed projection head during the next epoch.

Methods		KNN Accuracy
Fixed	No Projection	28.20
	Random Init	32.88
	DirectCLR [18]	33.08
	Pre-trained SimCLR	32.98
Moving	PCA top-128	32.19
	PCA bottom-128	31.58
	Slow-Single	33.06
	Slow-Optimal	32.29
Trainable	SimCLR [2]	38.44
	Ours	**40.48**

Table 2: *Experimental comparisons for KNN evaluation using different types of fixed, moving, and trainable linear projection head used for SimCLR pre-training on TinyImageNet dataset using ResNet-50 architecture. Note, both trainable versions of projection head consistently outperform all the fixed and slow moving alternatives and our proposed optimization scheme outperforms the standard SimCLR training procedure.*

- *Slow updates on Projection:* Another alternate optimization is via optimization of g in an optimization setting where g is optimized separately at the end of each epoch via optimization on the subset of dataset until convergence, namely Slow-Optimal or via optimization as a single step update via accumulated gradients of g during the training epoch of f, namely Slow-Single.

We present the experimental comparisons of KNN accuracies with above explained projection heads using SimCLR objective on TinyImageNet dataset in Table 2. Consistent to our observations in Table 1, our proposed optimization scheme performs better than standard SSL optimization scheme even for linear projection head. It is also clear from these comparisons that trainable projection head outperforms various forms of slow-moving or fixed projection heads.

In fact, our observations are contradictory to DirectCLR [18], which claims a fixed linear projection head can outperform a trainable linear projection head. Though, all the fixed projection head do outperform SimCLR trained model without projection, DirectCLR performs similar to the projection head setting where a fixed randomly initialized projection is employed. Interestingly, a pre-trained projection head obtained from SimCLR training performs significantly worse and roughly similar to the random fixed projection head. This clearly indicates that there is possibly no such fixed projection head that can outperform a trainable alternative. The worse performance for fixed pre-trained projection head can be accounted to the fact that for any fixed projection head g, the backbone network f can, in principle, be trained to achieve the same local minimum as without g. Thus, a fixed projection head is unable to aid in mitigating the shortcomings of contrastive loss.

Based on our observations, the linear projection head in most cases is low-rank. Thus, this simple implicit condition directs us towards a technically sound alternative, PCA as a replacement of the trainable projection head. Our analysis reveals that principal components computed at each epoch are worse than a fixed random initialization alternative. This further validates our hypothesis that a crucial condition on the projection head is to minimize the InfoNCE objective.

Our slow-moving version (Slow-Single) of alternate optimization indicates, that decoupling of update steps for backbone encoder and projection is beneficial and is thus able to achieve better performance than all the fixed projection head schemes. However, Slow-Optimal performs worse than Slow-Single which indicates a truncated optimization on projection head g is better than training to optimality.