Revisiting Contrastive Learning through the Lens of Neighborhood Component Analysis: an Integrated Framework

Ching-Yun Ko
MIT
`cyko@mit.edu`

Jeet Mohapatra
MIT
`jeetmo@mit.edu`

Sijia Liu
MSU
`liusiji5@msu.edu`

Pin-Yu Chen
IBM Research AI
`pin-yu.chen@ibm.com`

Luca Daniel
MIT
`luca@mit.edu`

Lily Weng
UCSD
`lweng@ucsd.edu`

Abstract

Contrastive learning aims to leverage pairs of positive and negative samples for representation learning. By investigating the connection between contrastive learning and neighborhood component analysis (NCA), we provide a novel stochastic nearest neighbor viewpoint of contrastive learning and subsequently propose a series of contrastive losses that outperform the existing ones. Under our proposed framework, we show a principled way to design integrated contrastive losses that simultaneously achieve good accuracy and robustness on downstream tasks. Long version is available at: `https://arxiv.org/abs/2112.04468`.

1 Introduction

The contrastive paradigm [1, 2, 3, 4, 5, 6] constructs an objective for embeddings based on an assumed semantic similarity, and the ability to distinguish dissimilar instances. When constructing the contrastive loss, contrastive learning algorithms [3, 4, 7, 6] typically build up the positive pairs by considering data augmentation $\mathcal{D}_x^{\text{aug}}$ of a data sample x due to the lack of label information. This heuristic loss [4] is denoted as $\mathcal{L}_{\text{SimCLR}}$. In this paper, we review $\mathcal{L}_{\text{SimCLR}}$ through the lens of the nearest neighbor classification in Neighborhood Component Analysis (NCA) [8]. Specifically, we uncover the relationship between stochastic nearest neighbors and positive pairs in contrastive learning, which then motivates a sequence of augmented contrastive losses that work better under practical computational constraints. A conceptual illustration of our proposals is given in Figure 1. Furthermore, by inspecting the adversarial accuracy of several methods (e.g., Figure 2's y-axis), one sees the insufficiency of those methods in addressing robustness. We thus present am integrated contrastive framework that accounts for *both* the standard and adversarial accuracy; this method's performance remains in the desired upper-right region (circled) as shown in Figure 2.

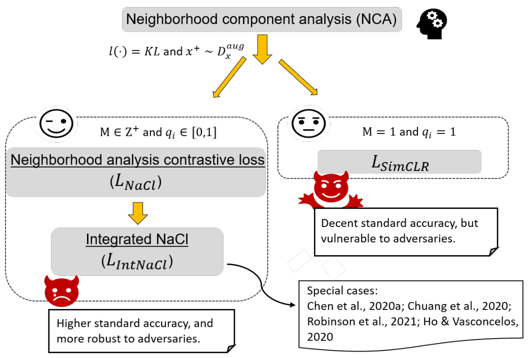
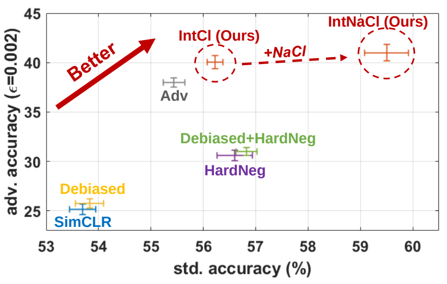
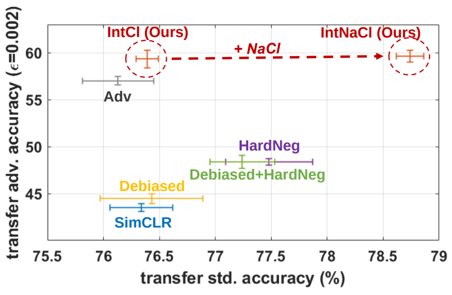

Neighborhood component analysis (NCA)
$l(\cdot) = KL$ and $x^+ \sim D_x^{aug}$
$M \in Z^+$ and $q_i \in [0,1]$
Neighborhood analysis contrastive loss
(L_{NaCl})
Integrated NaCl
($L_{IntNaCl}$)
Higher standard accuracy, and more robust to adversaries.
$M = 1$ and $q_i = 1$
L_{SimCLR}
Decent standard accuracy, but vulnerable to adversaries.
Special cases: Chen et al., 2020a; Chuang et al., 2020; Robinson et al., 2021; Ho & Vasconcelos, 2020

Figure 1: A conceptual illustration of the relationships among NCA, $\mathcal{L}_{\text{SimCLR}}$, and our proposals.



(a) CIFAR100



(b) CIFAR10

Figure 2: The performance of existing methods and our proposal (IntNaCl & IntCl) in terms of their standard accuracy (x-axis) and adversarial accuracy under FGSM attacks $\epsilon = 0.002$ (y-axis). The transfer performance refers to fine-tuning a linear layer for CIFAR10 with representation networks trained on CIFAR100.

2 Related Work

Getting rid of the memory bank [2, 3, 7] and instead makes use of other samples from the same batch to form contrastive pairs, SimCLR [4] still works under the noise contrastive estimation [9, 10, 11, 1] regime. We let $g_0(x, \{x_i^-\}_i^N)$ denote the negative term $\frac{1}{N}\sum_{i=1}^N e^{f(x)^T f(x_i^-)}$, where the subscript i identifies the summation index and the superscript N identifies the summation limits. We omit the subscript i when the sample index is one dimensional. Moreover, we define $K(A, B) = -\log(A/(A+B))$. Then $\mathcal{L}_{\text{SimCLR}}$ can be re-written as

$$\mathcal{L}_{\text{SimCLR}} := \mathbb{E}_{x\sim\mathcal{D}, x^+\sim\mathcal{D}_x^{\text{aug}}, x_i^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}}\left[K(e^{f(x)^T f(x^+)}, N g_0(x, \{x_i^-\}^N))\right]. \tag{1}$$

In [5], authors propose a *de-biased* constrastive loss to mitigate the sampling bias, which we denote as $\mathcal{L}_{\text{Debiased}}$. Then, [12] further propose to weigh sample pairs through the cosine distance in the estimator, and we denote their approach as $\mathcal{L}_{\text{Debiased+HardNeg}}$. These two losses differ from $\mathcal{L}_{\text{SimCLR}}$ in the estimator, where $\mathcal{L}_{\text{Debiased}}$ uses $g_1(x, \{u_i\}^n, \{v_j\}^m) = \max\{\frac{1}{1-\tau^+}(\frac{1}{n}\sum_{i=1}^n e^{f(x)^T f(u_i)} - \tau^+ \frac{1}{m}\sum_{j=1}^m e^{f(x)^T f(v_j)}), e^{-1/t}\}$ and $\mathcal{L}_{\text{Debiased+HardNeg}}$ uses $g_2(x, \{u_i\}^n, \{v_j\}^m) = \max\{\frac{1}{1-\tau^+}(\frac{\sum_{i=1}^n e^{(\beta+1)f(x)^T f(u_i)}}{\sum_{i=1}^n e^{\beta f(x)^T f(u_i)}} - \tau^+ \frac{1}{m}\sum_{j=1}^m e^{f(x)^T f(v_j)}), e^{-1/t}\}$. The n and m represents the numbers of sampled points in $\mathcal{D}_{\backslash x}^{\text{aug}}$ and $\mathcal{D}_x^{\text{aug}}$ for the weighted negative term, and τ^+ is a hyperparameter that encodes class prior.

In parallel to the above line of work, authors of [13] define the concept of *adversarial examples* in the regime of representation learning as the positive sample that maximizes $\mathcal{L}_{\text{SimCLR}}$ (i.e. Equation (1)) within a pre-specified perturbation magnitude. The resulting loss function is $\mathcal{L}_{\text{Adv}} = \mathbb{E}_{x\sim\mathcal{D}, x^+\sim\mathcal{D}_x^{\text{aug}}, x_{i_1}^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}, x_{i_2}^-\sim\mathcal{D}_{\backslash x}^{\text{adv}}}\Big[K(e^{f(x)^T f(x^+)}, N g_0(x, \{x_{i_1}^-\}^N)) + \alpha K(e^{f(x)^T f(x^{\text{adv}})}, N g_0(x, \{x_{i_2}^-\}^N))\Big]$, where the $\mathcal{D}_{\backslash x}^{\text{adv}}$ is defined by $\cup_{x'\in\mathcal{D}\backslash\{x\}} x' \cup x'^{,\text{adv}}$.

3 Neighborhood analysis Contrastive loss

Stochastic nearest neighbor framework. NCA is a supervised learning algorithm concerned with learning a distance metric that maximizes the performance of nearest neighbour classification. As the set of neighbors for a point can remain unchanged within an area around transformation A, the leave-one-out classification performance can be a piecewise-constant function of A and therefore non-differentiable. To overcome this, the optimization problem is generally given using the concept of stochastic nearest neighbors. In the stochastic nearest neighbor setting, nearest neighbor selection is regarded as a random event, where the probability point x_j is selected as the nearest neighbor for x_i is given as $p_{ij} = \frac{e^{-\|Ax_i - Ax_j\|^2}}{\sum_{k\neq i} e^{-\|Ax_i - Ax_k\|^2}} \propto e^{-\|Ax_i - Ax_j\|^2}$. Let c_i denote the label of x_i, in the leave-one-out classification loss, the probability a point is classified correctly is given as $p_i = \sum_{j|c_j=c_i} p_{ij}$, where $\{j \mid c_j = c_i\}$ defines an index set with which all points x_j belong to the same class as point x_i. We use M to denote the cardinality of this set. The probability x_i's label is c_i is given as q_i, which is exactly 1. Thus the optimization problem can be written as $\min_A \sum_{i=1}^n \ell(q_i, \sum_{j|c_j=c_i} p_{ij})$. This learning objective then naturally maximizes the expected accuracy of a 1-nearest neighbor classifier. We will focus on the KL divergence loss in this work. For $\ell(\cdot) = \text{KL}$, the relative entropy from p to q is $D_{\text{KL}}(q\|p) = \sum_i -q_i \log \frac{p_i}{q_i} = \sum_i -\log p_i$, when $q_i = 1$, and the optimization problem becomes $\min_A \sum_{i=1}^n -\log\left(\sum_{j|c_j=c_i} \frac{e^{-\|Ax_i - Ax_j\|^2}}{\sum_{k\neq i} e^{-\|Ax_i - Ax_k\|^2}}\right)$. With the above formulation, we argue that the contrastive learning loss can be given assuming only positive pairs belong to the same class and the transformation Ax is instead parametrized by a function $\frac{f(x)}{\sqrt{2}} = \frac{h(x)}{\sqrt{2}\|h(x)\|}$, where h is a neural network. Specifically, $\min_f \sum_{i=1}^n \; -\log\left(\sum_{j=1}^M \frac{e^{-\frac{1}{2}\left\|f(x_i)-f(x_j^+)\right\|^2}}{\sum_{k\neq i} e^{-\frac{1}{2}\|f(x_i)-f(x_k)\|^2}}\right) \xrightarrow{\|f(x)\|=1}$ $\min_f \mathbb{E}_{x\sim\mathcal{D}}\left[K(\sum_{j=1}^M e^{f(x)^T f(x_j^+)}, Ng_0(x, \{x_i^-\}^N))\right]$, which yields $\mathcal{L}_{\text{SimCLR}}$ with $M = 1, x^+ \sim \mathcal{D}_x^{\text{aug}}$.

NCA inspired contrastive loss. Although we have shown the connection between NCA and contrastive learning, applying the NCA framework exactly is challenging in two senses. On one hand, it requires us to use all possible negative pairs which is approximately the size of the entire dataset. Furthermore, to decide the "demographic " of a point's neighborhood, M depends on the relative density of positive to negative pairs one expects to have in the underlying data distribution. To tackle these, we propose to use a stochastic approximation to the population loss, where N is determined as a hyperparameter in a fashion similar to the batch size hyperparameter [4]. In order to determine M, as the expected relative density is task-dependent, we treat the M/N ratio as a hyperparameter.

Recall that by letting $\ell(\cdot) = \text{KL}$, $q_i = 1$, $M = 1, x^+ \sim \mathcal{D}_x^{\text{aug}}$, a general stochastic nearest neighbor algorithm yields $\mathcal{L}_{\text{SimCLR}}$. If we keep $\ell(\cdot) = \text{KL}$, $q_i = 1$, $M = 1$, then in practice, the expectation over the probability distribution $\mathcal{D}_x^{\text{aug}}$ is estimated by only one sample. To reduce the variance of such an estimator, with a slight abuse of notation, we denote the number of trials by M and propose to simulate M trials of the procedure for every x. This yields the following loss

$$\mathcal{L}_{\text{VAR}}(g = g_0, M) := \mathbb{E}_{x\sim\mathcal{D}, x_j^+\sim\mathcal{D}_x^{\text{aug}}, x_{ij}^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}}\left[\frac{1}{M}\sum_{j=1}^M K(e^{f(x)^T f(x_j^+)}, Ng_0(x, \{x_{ij}^-\}_i^N))\right].$$

By shifting our focus to the number of neighbors that are considered belonging to the same class (M), we admit the potential bias induced by assuming $M = 1$ (i.e. the relative density of positive to negative pairs to be $1/N$). Therefore, we experiment with enlarging the index set $\{j \mid c_j = c_i\}$ to include more than one element or equivalently $M \neq 1$. This leads us to the following objective

$$\mathcal{L}_{\text{BIAS}}(g = g_0, M) := \mathbb{E}_{x\sim\mathcal{D}, x_j^+\sim\mathcal{D}_x^{\text{aug}}, x_i^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}}\left[K(\sum_{j=1}^M e^{f(x)^T f(x_j^+)}, Ng_0(x, \{x_i^-\}^N))\right].$$

Finally, we challenge the specification of $q_i = 1$ and consider a synthetic data point $x' = \lambda x_i + (1-\lambda)y, y \sim \mathcal{D}$ that belongs to a synthetic class $c_{\lambda,i}$. Assume the probability x_i's label is $c_{\lambda,i}$ is $q_{\lambda,i} = \lambda + (1-\lambda)[c_y = c_i]$, then $q_{\lambda,i}$ should matches the probability $p_{\lambda,i} = \sum_{j|c_j=c_{\lambda,i}} p_{ij}$, where $\{j \mid c_j = c_{\lambda,i}\}$ is a singleton containing only the index of x', which yields

$$\begin{aligned}\mathcal{L}_{\text{MIXUP}}(g = g_0, M, \lambda) :=& \mathbb{E}_{x\sim\mathcal{D}, x^+\sim\mathcal{D}_x^{\text{aug}}, x_{i_1}^-, x_{i_2}^-, x_j^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}}\Big[K(e^{f(x)^T f(x^+)}, Ng_0(x, \{x_{i_1}^-\}^N)) \\ &+ \frac{\lambda}{M-1}\sum_{j=1}^{M-1} K(e^{f(x)^T f(\lambda x^+ + (1-\lambda)x_j^-)}, Ng_0(x, \{x_{i_2 j}^-\}_{i_2}^N)) \\ &+ \frac{1-\lambda}{M-1}\sum_{j=1}^{M-1} K(Ng_0(x, \{x_{i_2 j}^-\}_{i_2}^N), e^{f(x)^T f(\lambda x^+ + (1-\lambda)x_j^-)})\Big].\end{aligned}$$

The construction of x' assembles the mixup [14] in supervised learning. We therefore denote it by $\mathcal{L}_{\text{MIXUP}}$. As the above losses are designed from orthogonal perspectives, they are complementary to each other. We refer to the above three losses as **N**eighborhood **a**nalysis **C**ontrastive **l**oss (**NaCl**).

Table 1: The CIFAR100 linear evaluation results (%) of NaCl on $\mathcal{L}_{\text{SimCLR}}$, $\mathcal{L}_{\text{Debiased+HardNeg}}$, and $\mathcal{L}_{\text{IntCl}}$ (ours, cf. Equation (3)). The best improvement over the individual baseline is in boldface.

M	$\mathcal{L}_{\text{SimCLR}} : 53.69 \pm 0.25$			$\mathcal{L}_{\text{Debiased+HardNeg}} : 56.83 \pm 0.20$			$\mathcal{L}_{\text{IntCl}} : 56.22 \pm 0.15$		
	$\mathcal{L}_{\text{VAR}}$	$\mathcal{L}_{\text{BIAS}}$	$\mathcal{L}_{\text{MIXUP}}$	$\mathcal{L}_{\text{VAR}}$	$\mathcal{L}_{\text{BIAS}}$	$\mathcal{L}_{\text{MIXUP}}$	$\mathcal{L}_{\text{VAR}}$	$\mathcal{L}_{\text{BIAS}}$	$\mathcal{L}_{\text{MIXUP}}$
2	56.04±0.17	55.72±0.15	56.20±0.33	58.17±0.39	57.87±0.15	**60.69±2.43**	57.51±0.12	56.71±0.11	58.97±0.19
3	57.11±0.21	56.67±0.12	56.41±0.13	59.08±0.29	58.42±0.23	59.81±0.25	58.08±0.18	57.13±0.26	59.26±0.18
4	57.27±0.14	57.09±0.26	56.00±0.42	59.29±0.16	58.86±0.18	59.75±0.33	58.31±0.23	57.06±0.19	59.32±0.21
5	**57.91±0.12**	57.32±0.17	56.63±0.31	59.67±0.38	58.81±0.21	59.85±0.30	58.64±0.24	57.46±0.04	**59.43±0.23**

4 An integrated framework for contrastive learning

To design a generic loss that accounts for both clean and adversarial accuracy, together with a robustness-promoting term, we utilize the NaCl developed in Section 3 to construct a contrastive learning framework, called **Int**egrated **N**eighborhood **a**nalysis **C**ontrastive **l**oss (**IntNaCl**). A general form of $\mathcal{L}_{\text{IntNalL}}$ is given by

$$\mathcal{L}_{\text{IntNaCl}}(\alpha, \mathcal{L}_{\text{Na}}(g^1, M, \lambda), \mathcal{L}_{\text{Robust}}(g^2, w)) := \mathcal{L}_{\text{Na}}(g^1, M, \lambda) + \alpha \mathcal{L}_{\text{Robust}}(g^2, w), \tag{2}$$

where $\mathcal{L}_{\text{Na}}(g^1, M, \lambda)$ can be chose from $\{\mathcal{L}_{\text{VAR}}(g^1, M), \mathcal{L}_{\text{BIAS}}(g^1, M), \mathcal{L}_{\text{MIXUP}}(g^1, M, \lambda)\}$ and

$$\mathcal{L}_{\text{Robust}}(g^2, w) := \mathbb{E}\Big[K(e^{f(x)^T f(x^{\text{adv}})}, Ng^2(x, \cdot))w(x)\Big].$$

Variables g^1 and g^2 can be chose from $\{g_0, g_1, g_2\}$ and $w(x)$ facilitates goal-specific weighting scheme. Throughout our experiments, we will be using an adversarial weighting scheme $\hat{w}$ detailed in the appendix. Furthermore, we remark that as $\mathcal{L}_{\text{VAR}}$, $\mathcal{L}_{\text{BIAS}}$, and $\mathcal{L}_{\text{MIXUP}}$ all reduce to one same form when $M = 1$, we denote the $\mathcal{L}_{\text{IntNaCl}}$ under these cases by **Int**egrated **C**ontrastive **l**oss (**IntCl**):

$$\mathcal{L}_{\text{IntCL}}(\alpha, g^1, g^2, w) := \mathbb{E}\Big[K(e^{f(x)^T f(x^+)}, Ng^1(x, \cdot)) + \alpha K(e^{f(x)^T f(x^{\text{adv}})}, Ng^2(x, \cdot))w(x)\Big], \tag{3}$$

which reduces to $\mathcal{L}_{\text{SimCLR}}$ [4] with $\alpha = 0, g^1 = g_0$, to $\mathcal{L}_{\text{Debiased}}$ [5] with $\alpha = 0, g^1 = g_1$, to $\mathcal{L}_{\text{Debiased+HardNeg}}$ [12] with $\alpha = 0, g^1 = g_2$, to $\mathcal{L}_{\text{Adv}}$ [13] with $\alpha = 1, g^1 = g_0, g^2 = g_0, w(x) \equiv 1$.

5 Experimental results

In this section, we will evaluate three major properties of representation learning methods: standard downstream accuracy, transferability, and robustness. To evaluate the standard downstream accuracy, we train representation networks on CIFAR100 [15], freeze the network, and only fine-tune a fully-connected layer that maps representations to outputs on CIFAR100. To evaluate the transferability, we use the same representation networks and only fine-tune a fully-connected layer on CIFAR10.

Improvement over baselines. In this section, we test the effectiveness of NaCl in improving the downstream CIFAR100 classification accuracy. Specifically, we consider three baseline methods: 1) $\mathcal{L}_{\text{SimCLR}}$, or equivalently $\alpha = 0, g^1 = g_0, M = 1$ in Equation (2); 2) $\mathcal{L}_{\text{Debiased+HardNeg}}$, or equivalently $\alpha = 0, g^1 = g_2, M = 1$ in Equation (2); and 3) $\mathcal{L}_{\text{IntCl}}(\alpha = 1, g^1 = g_2, g^2 = g_2, w = \hat{w})$. We list the results in Table 1. Due to page limit, we only give the results of $\mathcal{L}_{\text{MIXUP}}$ with $\lambda = 0.9$ when applied on $\mathcal{L}_{\text{SimCLR}}$, and with $\lambda = 0.5$ when applied on $\mathcal{L}_{\text{Debiased+HardNeg}}$ and $\mathcal{L}_{\text{IntCl}}$. Complete tables of results obtained with $\lambda = 0.6, 0.7, 0.8$ can be found in the appendix. By referring to the Table 1, one can see that all three NaCl losses are able to improve the standard performance upon their baselines.

Robustness & Transferability. In addition to the discriminative power, we also want to empower the learned representation with strong adversarial performance and transferability. In appendix Section E, we list the classification accuracy on CIFAR100 under FGSM attacks with magnitude $\epsilon = 0.002$. From the table, one can see that, with the absence of robustness promoting loss ($\alpha = 0$), all NaCl methods manage to improve upon the baselines. Notably, $\mathcal{L}_{\text{MIXUP}}$ with $\lambda = 0.9$ boosts the CIFAR100 adversarial accuracy to 34.65% when applied to Debiased+HardNeg. That said, although $\mathcal{L}_{\text{VAR}}$ and $\mathcal{L}_{\text{BIAS}}$ are both useful in enhancing the adversarial performance, $\mathcal{L}_{\text{MIXUP}}$ improves the baseline by the largest margin. When we explicitly regularize the adversarial robustness performance ($\alpha \neq 0$), the representation network learned via $\mathcal{L}_{\text{IntCl}}$ yields an adversarial accuracy of 40.05%. When we strengthen the loss with NaCl, the adversarial performance is further improved.

We validate the transferability of all the methods by fine-tuning a fully-connected layer that maps representations to outputs on CIFAR10. This is in analogy to the evaluation procedure in [4] - train a fixed feature extractor on a large-scale dataset and train a linear classifier on top of the frozen base network with smaller-scale datasets. Section E in the appendix also shows that besides decent standard and adversarial performance, NCA inspired losses can also improve the transferability.

6 Conclusion

In this paper, we discover the relationship between contrastive loss and NCA, which motivates us to generalize existing contrastive losses to neighborhood analysis contrastive losses. We further propose a generic contrastive learning framework based on NaCl, which learns representations that score high in both standard accuracy and adversarial accuracy.

Funding Disclosure

This work was funded by the MIT-IBM collaboration under the research grant "Toward Trustworthy AI: Efficient Algorithms for Building Probably Robust and Verifiable Neural Networks".

References

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

[2] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *CVPR*, pp. 3733–3742, 2018.

[3] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *CVPR*, June 2020.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, vol. 119 of *Proceedings of Machine Learning Research*, (Virtual), pp. 1597–1607, PMLR, 13–18 Jul 2020.

[5] C.-Y. Chuang, J. Robinson, L. Yen-Chen, A. Torralba, and S. Jegelka, "Debiased contrastive learning," *arXiv preprint arXiv:2007.00224*, 2020.

[6] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent - a new approach to self-supervised learning," in *NeurIPS*, vol. 33, pp. 21271–21284, Curran Associates, Inc., 2020.

[7] X. Chen, H. Fan, R. B. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *CoRR*, vol. abs/2003.04297, 2020.

[8] J. Goldberger, G. E. Hinton, S. Roweis, and R. R. Salakhutdinov, "Neighbourhood components analysis," *NeurIPS*, vol. 17, pp. 513–520, 2004.

[9] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 297–304, JMLR Workshop and Conference Proceedings, 2010.

[10] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," in *ICML*, 2012.

[11] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu, "Exploring the limits of language modeling," *arXiv preprint arXiv:1602.02410*, 2016.

[12] J. D. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka, "Contrastive learning with hard negative samples," in *ICLR*, 2021.

[13] C.-H. Ho and N. Vasconcelos, "Contrastive learning with adversarial examples," *arXiv preprint arXiv:2010.12050*, 2020.

[14] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *ICLR*, 2018.

[15] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[16] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[17] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *ICLR*, 2014.

[18] A. Nguyen, J. Yosinski, and J. Clune, “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,” in *CVPR*, 2015.

[19] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” in *CVPR*, pp. 2574–2582, 2016.

[20] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *ICLR*, 2018.

[21] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial machine learning at scale,” in *ICLR*, 2016.

[22] H. Zeng, C. Zhu, T. Goldstein, and F. Huang, “Are adversarial examples created equal? a learnable weighted minimax risk for robustness under non-uniform attacks,” *arXiv preprint arXiv:2010.12989*, 2020.

[23] E. Wong, L. Rice, and J. Z. Kolter, “Fast is better than free: Revisiting adversarial training,” in *ICLR*, 2020.

[24] J. Z. HaoChen, C. Wei, A. Gaidon, and T. Ma, “Provable guarantees for self-supervised deep learning with spectral contrastive loss,” *arXiv preprint arXiv:2106.04156*, 2021.

Supplementary Materials

A Derivation from NCA to $\mathcal{L}_{\text{SimCLR}}$

$$
\begin{aligned}
&\arg\min_f \sum_{i=1}^{n} -\log\left(\sum_{j=1}^{M} \frac{e^{-\frac{1}{2}\left\|f(x_i)-f(x_j^+)\right\|^2}}{\sum_{k\neq i} e^{-\frac{1}{2}\|f(x_i)-f(x_k)\|^2}}\right)\\
=&\arg\min_f \sum_{i=1}^{n} -\log\left(\sum_{j=1}^{M} \frac{e^{f(x_i)^T f(x_j^+)-\frac{1}{2}\|f(x_i)\|^2-\frac{1}{2}\left\|f(x_j^+)\right\|^2}}{\sum_{k\neq i} e^{f(x_i)^T f(x_k)-\frac{1}{2}\|f(x_i)\|^2-\frac{1}{2}\|f(x_k)\|^2}}\right) && \text{(S1)}\\
=&\arg\min_f \sum_{i=1}^{n} -\log\left(\sum_{j=1}^{M} \frac{e^{f(x_i)^T f(x_j^+)-1}}{\sum_{k\neq i} e^{f(x_i)^T f(x_k)-1}}\right) && \text{(S2)}\\
=&\arg\min_f \sum_{i=1}^{n} -\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x_i)^T f(x_j^+)}}{\sum_{k\neq i} e^{f(x_i)^T f(x_k)}}\right)\\
=&\arg\min_f \sum_{i=1}^{n} -\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x_i)^T f(x_j^+)}}{\sum_{k\neq i, x_k\in\{x_j^+\}} e^{f(x_i)^T f(x_k)} + \sum_{k\neq i, x_k\notin\{x_j^+\}} e^{f(x_i)^T f(x_k)}}\right) && \text{(S3)}\\
=&\arg\min_f \mathbb{E}_{x\sim\mathcal{D}}\left[-\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^+)}}{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^+)} + \sum\limits_{i=1}^{N} e^{f(x)^T f(x_i^-)}}\right)\right] && \text{(S4)}\\
=&\arg\min_f \mathbb{E}_{x\sim\mathcal{D}}\left[-\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^+)}}{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^+)} + N g_0(x, \{x_i^-\}^N)}\right)\right],
\end{aligned}
$$

where we go from Equation (S1) to Equation (S2) based on the fact that $\|f(x)\| = 1$, and from Equation (S3) to Equation (S4) assuming that set $\{x_k : k \neq i\} = \{x_j^+ : 1 \leq j \leq M\} \cup \{x_i^- : 1 \leq i \leq N\}$.

B Adversarial Robustness.

Despite neural networks' supremacy in achieving impressive performance, they have been proved vulnerable to human-imperceptible perturbations [16, 17, 18, 19]. In the supervised learning setting, an adversarial perturbation δ is defined to render inconsistent classification result of the input x: $r(x+\delta) \neq r(x)$, where r is a neural network classifier. A stronger adversarial attack means it can find δ with higher success attack rate under the same ϵ-budget ($\|\delta\|_p \leq \epsilon$). One of the most popular and classical attack algorithms is FGSM [16], where with a fixed perturbation magnitude ϵ, FGSM uses the sign of cross entropy gradient to decide between $\delta = \epsilon$ and $\delta = -\epsilon$. Another popular attack method we consider in this paper is PGD [20], which assembles the iterative-FGSM [21] but with additional projection steps.

C Adversarial weighting

Weighting sample loss based on their margins has been proven to be effective in the adversarial training under supervised settings [22]. Specifically, it is argued that training points that are closer to the decision boundaries should be given more weight in the supervised loss. While the margin of a sample in supervised settings is well-defined, it is underdefined in unsupervised settings. To tackle this, we borrow the intelligence from [13] and mimic how the authors transfer the definition of adversarial examples in supervised learning to unsupervised learning. Specially, we see that as an adversarial example in supervised learning is defined by a perturbed sample that has a zero margin to the decision boundary, authors of [13] define adversarial example in unsupervised learning to be an augmented sample that maximizes the contrastive loss. With this, we also give our weighting function as the value of the contrastive loss $\hat{w}(x) := K(e^{f(x)^T f(x^+)}, Ng(x, \cdot))$, where the estimator g can be g_0, g_1, g_2. Using this, we see that samples that are originally hard to be distinguished from other samples (i.e. small probability) are now assigned with bigger weights.

D Experimental details

All the proposed methods are implemented based on open source repositories provided in the literature [4, 13, 12]. Five benchmarking contrastive losses are considered as baselines that include: $\mathcal{L}_{\text{SimCLR}}$ [4], $\mathcal{L}_{\text{Debiased}}$ [5], $\mathcal{L}_{\text{Debiased+HardNeg}}$ [12], $\mathcal{L}_{\text{Adv}}$ [13]. Unless otherwise specified, the representation network is trained for 100 epochs. We run five independent trials for each of the experiments and report the mean and standard deviation in the entries. Throughout our experiments, no adversarial fine-tuning is performed. We implement the proposed framework using PyTorch to enable the use of an NVIDIA GeForce RTX 2080 Super GPU, two NVIDIA Tesla P100 GPUs, and four NVIDIA Tesla V100 GPUs.

Architecture. We follow [4, 12] to incorporate an MLP projection head during the contrastive learning on resnet18.

Optimizer. Adam optimizer with a learning rate of $3e-4$.

Batching. A batch size of 256 is used across all the experiments.

Methodological hyperparameters. Throughout out experiments, we use $\tau^+ = 0.01$ and $\beta = 1.0$ for $\mathcal{L}_{\text{Debiased}}$ [5] and $\mathcal{L}_{\text{Debiased+HardNeg}}$ [12], $\alpha = 1$ for $\mathcal{L}_{\text{Adv}}$ [13]. The same set of hyperparameters are used in our IntCl and IntNaCl.

Data augmentation. Our data augmentation includes random resized crop, random horizontal flip, random grayscale, and color jitter. Specifically, we implement the color jitter by calling $torchvision.transforms.ColorJitter(0.8*s, 0.8*s, 0.8*s, 0.2*s)$ and execute with probability 0.8. Random grayscale is performed with probability 0.2.

Adversarial hyperparameters. When evaluating the adversarial robustness using the codebase provided in [23], we use a PGD step size of $1e-2$, 10 iterations, and 2 random restarts.

Error bar. We run five independent trials for each of the experiments and report the mean and standard deviation for all tables and figures. The error bars in Figure S1 is omitted for better visual clarity.

E Complete tables of results

We give the full table of results in Section 5 in the following. Notably, we gather the standard accuracy, adversarial accuracy, transfer accuracy, and transfer adversarial accuracy for each specification.

Table S1: The effectiveness evaluation of NaCl on SimCLR (i.e. $\alpha = 0, g^1 = g_0$). The best performance within each loss type is in boldface. We color the overall best performance in blue.

M	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_0, M, \lambda) = \mathcal{L}_{\text{VAR}}(g_0, M)$			
	CIFAR100 Acc.	FGSM Acc.	CIFAR10 Acc.	FGSM Acc.
1	53.69±0.25	25.17±0.55	76.34±0.28	43.50±0.41
2	56.04±0.17	27.19±0.79	77.32±0.14	**44.61±0.33**
3	57.11±0.21	27.39±0.36	78.02±0.27	44.23±0.39
4	57.27±0.14	27.63±0.78	77.91±0.29	42.97±0.61
5	**57.91±0.12**	**28.37±0.56**	**78.09±0.29**	44.51±0.44
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_0, M, \lambda) = \mathcal{L}_{\text{BIAS}}(g_0, M)$			
1	53.69±0.25	25.17±0.55	76.34±0.28	43.50±0.41
2	55.72±0.15	27.04±0.45	77.40±0.14	44.58±0.41
3	56.67±0.12	**28.41±0.24**	77.53±0.24	**45.21±0.89**
4	57.09±0.26	28.20±0.81	77.75±0.22	45.13±0.44
5	**57.32±0.17**	28.33±0.59	**77.93±0.40**	44.46±0.53
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_0, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_0, M, 0.5)$			
1	53.69±0.25	**25.17±0.55**	76.34±0.28	**43.50±0.41**
2	54.76±0.29	23.66±0.27	76.78±0.26	40.76±0.66
3	55.21±0.17	24.46±0.44	77.45±0.18	41.78±0.80
4	55.68±0.27	24.19±0.46	77.40±0.24	41.33±0.34
5	**55.85±0.16**	24.01±0.91	**77.50±0.16**	40.77±0.66
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_0, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_0, M, 0.6)$			
1	53.69±0.25	25.17±0.55	76.34±0.28	**43.50±0.41**
2	54.84±0.35	25.94±0.81	77.11±0.15	42.81±0.83
3	55.49±0.13	**26.25±0.89**	76.95±0.32	42.99±0.96
4	55.65±0.24	25.41±0.53	**77.39±0.37**	42.69±1.20
5	**55.66±0.22**	26.01±0.60	77.26±0.48	43.06±0.79
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_0, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_0, M, 0.7)$			
1	53.69±0.25	25.17±0.55	76.34±0.28	43.50±0.41
2	55.57±0.32	27.67±0.60	77.09±0.27	44.68±0.71
3	55.83±0.25	27.72±0.59	77.23±0.28	43.68±0.72
4	56.29±0.25	**27.92±0.60**	77.33±0.29	44.69±0.82
5	**56.37±0.32**	27.78±0.54	**77.40±0.20**	**45.07±0.98**
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_0, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_0, M, 0.8)$			
1	53.69±0.25	25.17±0.55	76.34±0.28	43.50±0.41
2	55.75±0.21	29.30±0.86	76.80±0.20	46.56±1.02
3	56.27±0.26	**29.96±0.29**	77.11±0.37	46.52±0.50
4	**56.39±0.26**	29.49±0.65	77.34±0.31	46.79±0.93
5	56.23±0.13	29.47±0.95	**77.40±0.14**	**47.36±0.69**
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_0, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_0, M, 0.9)$			
1	53.69±0.25	25.17±0.55	76.34±0.28	43.50±0.41
2	56.20±0.33	30.95±0.36	76.96±0.15	**48.85±0.75**
3	56.41±0.13	**30.98±0.90**	77.10±0.21	48.76±0.63
4	56.00±0.42	29.90±0.63	**77.11±0.40**	48.16±0.40
5	**56.63±0.31**	30.58±0.52	77.04±0.19	47.96±0.46

Table S2: The effectiveness evaluation of NaCl on Debised+HardNeg (i.e. $\alpha = 0, g^1 = g_2$). The best performance within each loss type is in boldface. We color the overall best performance in blue.

M	CIFAR100 Acc.	FGSM Acc.	CIFAR10 Acc.	FGSM Acc.
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{VAR}}(g_2, M)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	**48.38±0.70**
2	58.17±0.39	31.92±0.45	77.43±0.18	48.05±0.38
3	59.08±0.29	32.63±0.74	77.87±0.29	47.58±0.57
4	59.29±0.16	32.48±0.62	77.92±0.17	47.08±0.53
5	**59.67±0.38**	**33.10±0.71**	**78.04±0.09**	46.90±0.91
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{BIAS}}(g_2, M)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	57.87±0.15	32.50±0.48	77.43±0.11	48.14±0.31
3	58.42±0.23	**33.19±0.60**	77.41±0.17	48.09±0.93
4	**58.86±0.18**	32.65±1.07	77.46±0.29	**48.43±0.94**
5	58.81±0.21	32.86±0.47	**77.58±0.23**	48.30±0.39
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.5)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	**60.69±2.43**	**32.22±0.35**	79.36±0.65	48.86±0.34
3	59.81±0.25	32.04±0.67	79.41±0.17	48.91±0.81
4	59.75±0.33	32.03±0.34	79.42±0.18	**49.05±0.71**
5	59.85±0.30	32.06±0.72	**79.45±0.20**	48.32±0.70
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.6)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	58.94±0.29	32.65±0.36	78.67±0.15	**49.86±0.59**
3	59.43±0.35	32.91±0.40	78.94±0.19	48.84±1.09
4	**59.54±0.28**	33.02±0.62	78.92±0.29	49.64±0.74
5	59.52±0.28	**33.10±0.50**	**79.29±0.21**	49.39±1.02
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.7)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	58.24±0.19	33.24±0.90	78.30±0.31	**50.40±0.83**
3	58.74±0.26	33.12±0.59	78.49±0.30	49.85±0.38
4	58.79±0.38	**33.63±0.53**	78.51±0.29	49.88±0.75
5	**58.99±0.18**	32.93±0.81	**78.57±0.12**	49.53±1.55
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.8)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	57.60±0.15	34.14±0.22	**77.96±0.07**	**51.82±0.68**
3	58.04±0.28	33.93±0.45	77.55±0.18	50.30±0.81
4	58.05±0.16	**34.16±0.54**	77.90±0.21	50.40±0.43
5	**58.43±0.27**	33.87±0.62	77.90±0.17	50.78±0.95
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.9)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	57.16±0.15	34.25±0.55	77.19±0.09	**51.42±0.45**
3	57.08±0.10	33.96±0.19	77.21±0.26	51.30±1.05
4	57.36±0.19	**34.29±0.15**	**77.34±0.34**	51.16±0.55
5	**57.38±0.16**	34.25±0.30	77.13±0.16	50.68±0.74

Table S3: The effectiveness evaluation of NaCl ($M \neq 1$) on IntCl ($M = 1$) when $\alpha = 1, g^1 = g^2 = g_2$. The best performance within each loss type is in boldface. We color the overall best performance in blue.

M	CIFAR100 Acc.	FGSM Acc.	CIFAR10 Acc.	FGSM Acc.
	$\alpha \neq 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{VAR}}(g_2, M)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	**59.33±0.94**
2	57.51±0.12	41.01±0.36	76.88±0.49	58.77±0.67
3	58.08±0.18	41.02±0.83	76.95±0.19	58.28±0.50
4	58.31±0.23	**41.49±0.51**	77.30±0.30	58.61±0.80
5	**58.64±0.24**	40.50±0.23	**77.42±0.17**	58.11±0.72
	$\alpha \neq 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{BIAS}}(g_2, M)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	**59.33±0.94**
2	56.71±0.11	39.80±0.57	76.55±0.27	58.44±0.31
3	57.13±0.26	40.53±0.29	**76.67±0.22**	58.47±0.31
4	57.06±0.19	40.85±0.31	76.34±0.22	58.91±0.62
5	**57.46±0.04**	**41.00±0.86**	76.60±0.37	57.98±0.47
	$\alpha \neq 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.5)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	59.33±0.94
2	58.97±0.19	40.25±0.52	78.61±0.20	58.41±0.59
3	59.26±0.18	40.96±0.58	**78.83±0.22**	59.20±1.25
4	59.32±0.21	40.82±0.54	78.83±0.27	59.03±0.52
5	**59.43±0.23**	**41.01±0.34**	78.80±0.21	**59.51±0.93**
	$\alpha \neq 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.6)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	59.33±0.94
2	58.55±0.34	**40.85±0.62**	78.34±0.22	**59.56±0.88**
3	59.05±0.21	40.83±0.44	78.41±0.12	59.14±0.78
4	59.06±0.25	40.80±0.89	78.61±0.22	58.41±1.00
5	**59.10±0.23**	40.68±0.50	**78.63±0.21**	58.92±0.76
	$\alpha \neq 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.7)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	59.33±0.94
2	58.00±0.18	40.35±0.34	77.73±0.24	59.40±1.27
3	58.23±0.18	40.94±0.75	77.91±0.25	**59.57±0.81**
4	58.20±0.25	40.95±0.45	77.89±0.20	59.49±0.49
5	**58.37±0.14**	**41.15±0.48**	**78.27±0.26**	59.17±0.94
	$\alpha \neq 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.8)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	59.33±0.94
2	57.07±0.24	41.29±0.57	77.27±0.28	60.16±0.51
3	**57.62±0.22**	40.93±0.49	77.54±0.27	59.47±0.52
4	57.61±0.25	**41.36±0.41**	77.50±0.34	**60.28±0.68**
5	57.56±0.18	40.71±0.34	**77.58±0.42**	59.99±0.30
	$\alpha \neq 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.9)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	59.33±0.94
2	56.54±0.33	40.85±0.13	76.81±0.22	60.40±0.46
3	56.69±0.11	41.23±0.66	**76.98±0.22**	60.13±0.56
4	56.43±0.26	**41.56±0.56**	76.97±0.20	**61.21±0.49**
5	**56.86±0.11**	41.09±0.31	76.91±0.21	60.09±0.39

F Adversarial accuracy

For a more comprehensive study of adversarial robustness, we extend Table S2 to include PGD attack results with the same strength as FGSM attacks ($\epsilon = 0.002$). One can readily see from Table S4 that the adversarial accuracy under PGD attacks of the same magnitude is slightly lower (roughly 2-3% lower) as PGD is a stronger attack. Nevertheless, the trend is consistent – the models that exhibit better adversarial robustness w.r.t. FGSM attacks also demonstrate superior adversarial robustness w.r.t. PGD attacks.

Table S4: The complete Table S2 (Table 1 right column) with additional PGD accuracy.

M	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{VAR}}(g_2, M)$					
	CIFAR100 Acc.	FGSM Acc.	PGD Acc.	CIFAR10 Acc.	FGSM Acc.	PGD Acc.
1	56.83±0.20	31.03±0.41	28.80±0.48	77.24±0.29	**48.38±0.70**	**46.24±0.77**
2	58.17±0.39	31.92±0.45	29.77±0.43	77.43±0.18	48.05±0.38	45.63±0.50
3	59.08±0.29	32.63±0.74	30.33±0.84	77.87±0.29	47.58±0.57	45.02±0.62
4	59.29±0.16	32.48±0.62	30.12±0.73	77.92±0.17	47.08±0.53	44.52±0.54
5	**59.67±0.38**	**33.10±0.71**	**30.87±0.88**	**78.04±0.09**	46.90±0.91	44.20±1.08
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{BIAS}}(g_2, M)$					
1	56.83±0.20	31.03±0.41	28.80±0.48	77.24±0.29	48.38±0.70	**46.24±0.77**
2	57.87±0.15	32.50±0.48	30.25±0.60	77.43±0.11	48.14±0.31	45.81±0.43
3	58.42±0.23	**33.19±0.60**	**30.93±0.59**	77.41±0.17	48.09±0.93	45.67±0.93
4	**58.86±0.18**	32.65±1.07	30.22±1.09	77.46±0.29	**48.43±0.94**	45.99±1.15
5	58.81±0.21	32.86±0.47	30.57±0.55	**77.58±0.23**	48.30±0.39	45.80±0.48
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.5)$					
1	56.83±0.20	31.03±0.41	28.80±0.48	77.24±0.29	48.38±0.70	46.24±0.77
2	**60.69±2.43**	**32.22±0.35**	**30.11±0.43**	79.36±0.65	48.86±0.34	46.67±0.40
3	59.81±0.25	32.04±0.67	29.87±0.65	79.41±0.17	48.91±0.81	46.61±0.86
4	59.75±0.33	32.03±0.34	29.85±0.36	79.42±0.18	**49.05±0.71**	**46.70±0.80**
5	59.85±0.30	32.06±0.72	29.99±0.76	**79.45±0.20**	48.32±0.70	45.89±0.82
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.6)$					
1	56.83±0.20	31.03±0.41	28.80±0.48	77.24±0.29	48.38±0.70	46.24±0.77
2	58.94±0.29	32.65±0.36	30.16±0.27	78.67±0.15	**49.86±0.59**	**47.38±0.70**
3	59.43±0.35	32.91±0.40	30.36±0.52	78.94±0.19	48.84±1.09	46.24±1.32
4	**59.54±0.28**	33.02±0.62	**30.68±0.72**	78.92±0.29	49.64±0.74	47.15±0.88
5	59.52±0.28	**33.10±0.50**	30.63±0.48	**79.29±0.21**	49.39±1.02	46.89±1.12
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.7)$					
1	56.83±0.20	31.03±0.41	28.80±0.48	77.24±0.29	48.38±0.70	46.24±0.77
2	58.24±0.19	33.24±0.90	30.40±1.06	78.30±0.31	**50.40±0.83**	**47.50±0.89**
3	58.74±0.26	33.12±0.59	29.94±0.62	78.49±0.30	49.85±0.38	46.69±0.32
4	58.79±0.38	**33.63±0.53**	**30.70±0.60**	78.51±0.29	49.88±0.75	47.01±0.96
5	**58.99±0.18**	32.93±0.81	29.89±0.99	**78.57±0.12**	49.53±1.55	46.41±1.91
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.8)$					
1	56.83±0.20	31.03±0.41	28.80±0.48	77.24±0.29	48.38±0.70	46.24±0.77
2	57.60±0.15	34.14±0.22	31.35±0.25	**77.96±0.07**	**51.82±0.68**	**48.81±0.85**
3	58.04±0.28	33.93±0.45	31.31±0.62	77.55±0.18	50.30±0.81	47.41±0.76
4	58.05±0.16	**34.16±0.54**	**31.41±0.61**	77.90±0.21	50.40±0.43	47.58±0.47
5	**58.43±0.27**	33.87±0.62	31.23±0.76	77.90±0.17	50.78±0.95	47.96±1.12
	$\alpha = 0,\ \mathcal{L}_{\text{Na}}(g_2, M, \lambda) = \mathcal{L}_{\text{MIXUP}}(g_2, M, 0.9)$					
1	56.83±0.20	31.03±0.41	28.80±0.48	77.24±0.29	48.38±0.70	46.24±0.77
2	57.16±0.15	34.25±0.55	31.83±0.57	77.19±0.09	**51.42±0.45**	**49.09±0.53**
3	57.08±0.10	33.96±0.19	31.56±0.34	77.21±0.26	51.30±1.05	48.60±1.28
4	57.36±0.19	**34.29±0.15**	**31.93±0.32**	**77.34±0.34**	51.16±0.55	48.64±0.61
5	**57.38±0.16**	34.25±0.30	31.89±0.26	77.13±0.16	50.68±0.74	48.14±0.83

In Figure S1, we show the adversarial accuracy as a function of the FGSM attack strength ϵ. Specifically, we range the attack strength from 0.002 to 0.032 and give the adversarial accuracy of our proposals (IntCl & IntNaCl) together with baselines under all attacks. From Figure S1, one can see that among all baselines, AdvW demonstrates the best adversarial robustness, whereas our proposals still consistently win over it by a noticeable margin.

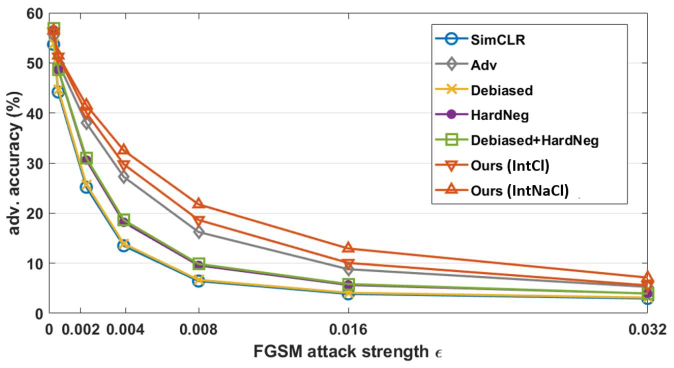


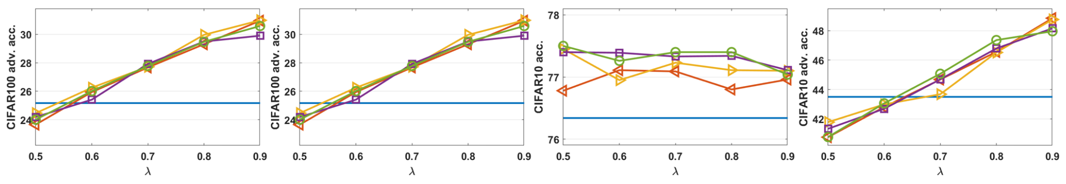
Figure S1: The adversarial accuracy under FGSM attacks of different strength on CIFAR100.

G Accuracy on larger epochs.

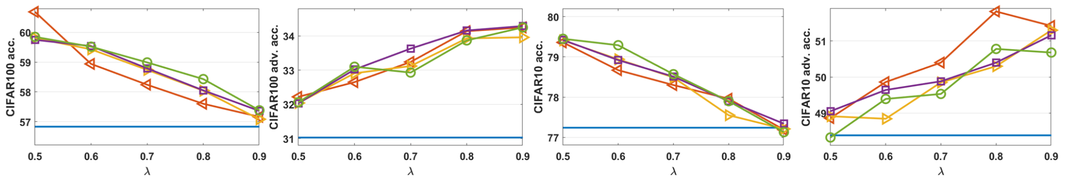
As training the representation with more epochs can also expose the data to more augmentations, we carry out an additional experiments to compare the efficiency and effectiveness of baseline methods with significant more training epochs. Specially, [24] has reported a $\mathcal{L}_{\text{SimCLR}}$ CIFAR100 accuracy of 54.74% after 200 epochs, compared to $\mathcal{L}_{\text{VAR}}(g_0, 2)$'s 56.04% after 100 epochs. In our reproduction of the $\mathcal{L}_{\text{SimCLR}}$ 200-epoch result, we have witnessed an accuracy of 57.45% however at the cost of 1.34X training time (cf. 200 epochs with $\mathcal{L}_{\text{SimCLR}}$ takes 211 mins vs. 100 epochs with $\mathcal{L}_{\text{VAR}}(g_0, 2)$ takes 158 mins). Additionally, we see $\mathcal{L}_{\text{SimCLR}}$ reaches 61.90% and Debiased+HardNeg stops at 62.74%, while $\mathcal{L}_{\text{VAR}}(g_0, 2)$ and $\mathcal{L}_{\text{VAR}}(g_2, 2)$ improve upon them individually by reaching 62.37% and 63.51%. We refer the readers to the appendix for detailed linear evaluation results on CIFAR100 with extended training epochs.

H The effect of λ

To investigate the effect of λ on different metrics, we include in Figure S2 the standard and adversarial accuracy on CIFAR100 and CIFAR10 as functions of λ. Intriguingly, we see that all the accuracy curves in Figure S2(a) have tended to increase over λ. Comparatively, two of the accuracy curves in Figure S2(b), the standard accuracy on CIFAR100 and CIFAR10, show downward trends. One plausible reason is related to the room for improvements of individual baselines. Since Debiased+HardNeg is a much stronger baseline than SimCLR, it is closer to the robustness-accuracy trade-off.



(a) NaCl on SimCLR, i.e. $\alpha = 0, \mathcal{L}_{\text{Na}} = \mathcal{L}_{\text{MIXUP}}, g^1 = g_0$ in Equation (2)



(b) NaCl on Debiased+HardNeg, i.e. $\alpha = 0, \mathcal{L}_{\text{Na}} = \mathcal{L}_{\text{MIXUP}}, g^1 = g_2$ in Equation (2)

Figure S2: The standard and adversarial accuracy (%) on CIFAR100 and CIFAR10 as functions of λ.

I Extended runtime

#epoch	100	200	400	600	800	1000
$\mathcal{L}_{\text{SimCLR}}$	53.69	57.45	60.06	60.96	61.27	**61.90**
$\mathcal{L}_{\text{VAR}}(g_0, 2)$	56.04	59.44	61.42	**62.37**	62.06	-
$\mathcal{L}_{\text{BIAS}}(g_0, 2)$	55.72	59.31	61.19	61.66	**62.49**	-
$\mathcal{L}_{\text{MIXUP}}(g_0, 2, 0.9)$	56.20	58.98	61.81	62.43	**62.46**	-
$\mathcal{L}_{\text{Debiased+HardNeg}}$	56.83	59.35	61.77	**62.74**	62.68	-
$\mathcal{L}_{\text{VAR}}(g_2, 2)$	58.17	60.66	62.38	63.43	**63.51**	-
$\mathcal{L}_{\text{BIAS}}(g_2, 2)$	57.87	60.06	62.36	62.58	**62.86**	-
$\mathcal{L}_{\text{MIXUP}}(g_2, 2, 0.5)$	60.69	62.14	64.06	**65.59**	65.53	-

Table S5: The CIFAR100 linear evaluation results (%) after different numbers of training epochs.

J Supervised learning baseline

We give in the following the standard and adversarial accuracy of a supervised learning baseline with the same network architecture, optimizer, and batch size. In our self-supervised representation learning experiments, we train the representation network for 100 epochs and train the downstream fully-connected classifying layer for 1000 epochs. Therefore, to obtain a fair supervised learning baseline, we train the complete network end-to-end for 1000 epochs. We follow the same procedures in evaluating the transfer standard accuracy and adversarial accuracy as described in Section 5.

CIFAR100 (std. acc., FGSM acc., PGD acc.): 65.16 ± 0.32, 35.89 ± 0.23, 32.62 ± 0.23.

Transfer CIFAR10 (std. acc., FGSM acc., PGD acc.): 77.45 ± 0.21, 44.39 ± 0.47, 40.35 ± 0.52.