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

Ching-Yun Ko [1] Jeet Mohapatra [1] Sijia Liu [2] Pin-Yu Chen [3] Luca Daniel [1] Tsui-Wei Weng [4]

Abstract

As a seminal tool in self-supervised representation learning, contrastive learning has gained unprecedented attention in recent years. In essence, contrastive learning aims to leverage pairs of positive and negative samples for representation learning, which relates to exploiting neighborhood information in a feature space. 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 new methodology to design integrated contrastive losses that could simultaneously achieve good accuracy and robustness on downstream tasks. With the integrated framework, we achieve up to 6% improvement on the standard accuracy and 17% improvement on the robust accuracy.

1. Introduction

Contrastive learning has drawn much attention and has become one of the most effective representation learning techniques recently. The contrastive paradigm (Oord et al., 2018; Wu et al., 2018; He et al., 2020; Chen et al., 2020a; Chuang et al., 2020; Grill et al., 2020) constructs an objective for embeddings based on an assumed semantic similarity between positive pairs and dissimlarity between negative pairs, which stems from instance-level classification (Dosovitskiy et al., 2015; Bojanowski & Joulin, 2017; Wu et al., 2018). Specifically, the contrastive loss $\mathcal{L}_{\text{CL}}$ (Oord et al., 2018; Chen et al., 2020a) is defined as $\mathbb{E}_{\substack{x\sim\mathcal{D},\\ x^+\sim\mathcal{D}_x^+,\\ x_i^-\sim\mathcal{D}_x^-}}\left[-\log\frac{e^{f(x)^Tf(x^+)}}{e^{f(x)^Tf(x^+)}+\sum_{i=1}^{N}e^{f(x)^Tf(x_i^-)}}\right]$ where, for an input data sample x, (x,x^+) denotes a positive pair and (x,x^-) denotes a negative pair. The function f is an encoder parameterized by a neural network and the number of negative pairs N is typically treated as a hyperparameter. Note that the contrastive loss can encode the inputs and keys by different encoders if one considers the use of memory bank or momentum contrast (Wu et al., 2018; He et al., 2020; Chen et al., 2020b). In this work, we will focus on the paradigm proposed in (Wang & Gupta, 2015; Ye et al., 2019; Chen et al., 2020a) which has demonstrated competitive results in representation learning.

When constructing loss $\mathcal{L}_{\text{CL}}$, ideally, one draws x^+ from the data distribution $\mathcal{D}_x^+$ that characterizes the semantically-*similar* (i.e., *positive*) samples to x; similarly, one wants to draw x^- from $\mathcal{D}_x^-$ that characterizes the semantically-*dissimilar* (*negative*) samples. However, the definition of semantically-*similar* and semantically-*dissimilar* is heavily contingent on downstream tasks: an image of a cat can be considered semantically similar to that of a dog if the downstream task is to distinguish between animal and non-animal classes. Without the knowledge of downstream tasks, $\mathcal{D}_x^+$ and $\mathcal{D}_x^-$ are hard to define. To provide a surrogate of measuring similarity, current mainstream contrastive learning algorithms (He et al., 2020; Chen et al., 2020a;b; Grill et al., 2020) typically build up $\mathcal{D}_x^+$ by considering data augmentation $\mathcal{D}_x^{\text{aug}}$ of a data sample x. In the meantime, $\mathcal{D}_x^-$ is approximated by the joint distribution $\mathcal{D}$ or $\mathcal{D}_{\setminus x}^{\text{aug}} := \cup_{x'\in\mathcal{D}\setminus\{x\}}\mathcal{D}_{x'}^{\text{aug}}$, and the resulting contrastive loss is known as $\mathcal{L}_{\text{SimCLR}}$ which was proposed in (Chen et al., 2020a):

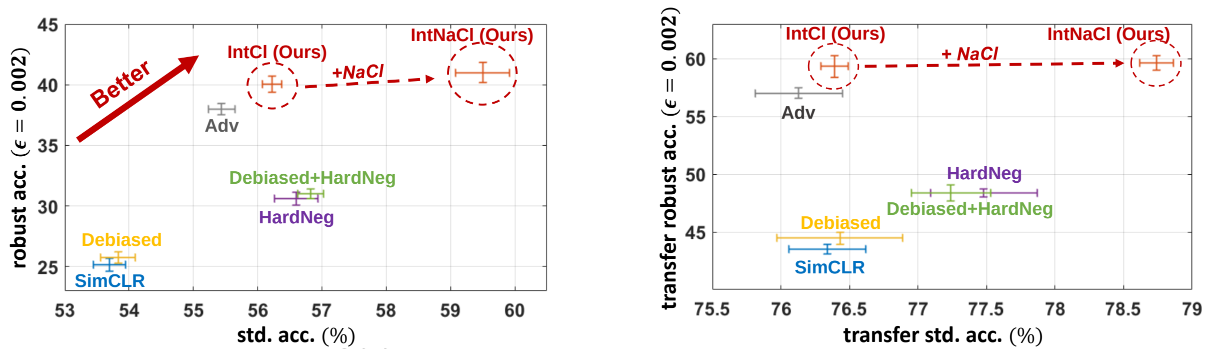
$$(\text{SimCLR loss } \mathcal{L}_{\text{SimCLR}})$$

$$\mathbb{E}_{\substack{x\sim\mathcal{D},\\ x^+\sim\mathcal{D}_x^{\text{aug}},\\ x_i^-\sim\mathcal{D}_{\setminus x}^{\text{aug}}}}\left[-\log\frac{e^{f(x)^Tf(x^+)}}{e^{f(x)^Tf(x^+)}+\sum_{i=1}^{N}e^{f(x)^Tf(x_i^-)}}\right]. \quad (1)$$

Although this formulation seems to put no assumptions on the downstream task classes, we find that there are in fact implicit assumptions on the class probability prior of the downstream tasks. Specifically, we formally establish the connection between the Neighborhood Component Analysis (NCA) and the unsupervised contrastive learning in this

[1]MIT [2]MSU [3]IBM Research AI [4]UCSD. Correspondence to: Ching-Yun Ko <cyko@mit.edu>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).



(a) CIFAR100 (b) CIFAR10

Figure 1. The performance of existing methods and our proposal (IntNaCl & IntCl) in terms of their standard accuracy (x-axis) and robust 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.

paper for the first time (to our best knowledge). Inspired by this interesting relationship to NCA, we further propose two new contrastive loss (named NaCl) which outperform existing paradigm. Furthermore, by inspecting the robust accuracy of several existing methods (e.g., Figure 1's y-axis, the classification accuracy when inputs are corrupted by crafted perturbations), one can see the insufficiency of existing methods in addressing robustness. Thus, we propose a new integrated contrastive framework (named IntNacl and IntCl) that accounts for *both* the standard accuracy and adversarial cases: our proposed method's performance remains in the desired upper-right region (circled) as shown in Figure 1.

We summarize our main contributions as follows:

- We establish the relationship between contrastive learning and NCA, and propose two new contrastive loss dubbed **NaCl** (Neighborhood analysis Contrastive loss). We provide theoretical analysis on NaCl and show better generalization bounds over the baselines;
- Building on top of NaCl, we propose a generic framework called Integrated contrastive learning (**IntCl** and **IntNaCl**) where we show that the spectrum of recently-proposed contrastive learning losses (Chuang et al., 2020; Robinson et al., 2021; Ho & Vasconcelos, 2020) can be included as special cases of our framework;
- We provide extensive experiments that demonstrate the effectiveness of IntNaCl in improving standard accuracy and robust accuracy. Specifically, IntNaCl improves upon literature (Chen et al., 2020a; Chuang et al., 2020; Robinson et al., 2021; Ho & Vasconcelos, 2020) by 3-6% and 4-16% in CIFAR100 standard and robust accuracy, and 2-3% and 3-17% in CIFAR10 standard and robust accuracy, respectively.

2. Related Work and Preliminaries

Contrastive learning. In the early work of (Dosovitskiy et al., 2015), authors treat every individual image in a dataset as belonging to its own class and do multi-class classification tasks under the setting. However, this regime will soon become intractable as the size of dataset increases. To cope with this, (Wu et al., 2018) designs a memory bank for storing seen representations (keys) and utilize noise contrastive estimation (Gutmann & Hyvärinen, 2010; Mnih & Teh, 2012; Jozefowicz et al., 2016; Oord et al., 2018) for representation comparisons. (He et al., 2020) and (Chen et al., 2020b) further improve upon (Wu et al., 2018) by storing keys inferred from a momentum encoder other than the representation encoder for x. To further reduce the computational cost, besides the practical tricks introduced in SimCLR (Chen et al., 2020a) (e.g. stronger data augmentation scheme and projector heads), authors of SimCLR get rid of the memory bank and instead makes use of other samples from the same batch to form contrastive pairs.

In the rest of this paper, we will focus on the setups of SimCLR and the related follow up work (Chuang et al., 2020; Robinson et al., 2021; Ho & Vasconcelos, 2020) due to computational efficiency. A temperature scaling hyperparameter t is normally used in contrastive learning to tune the radius of the hypersphere that representations lie in. For better readability, without loss of generality, we let $t = 1$ in all equations. We let $g_0(x, \{x_i^-\}^N)$ 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 (e.g. x_i^- has 1-D index, x_{ij}^- has 2-D index). Then $\mathcal{L}_{\text{SimCLR}}$ in Equation (1) can be re-written as

(Re-written SimCLR loss $\mathcal{L}_{\text{SimCLR}}$)

$$\mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D},\\ x^+\sim\mathcal{D}_x^{\text{aug}},\\ x_i^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}}}\left[-\log\frac{e^{f(x)^Tf(x^+)}}{e^{f(x)^Tf(x^+)}+Ng_0(x,\{x_i^-\}^N)}\right]. \quad (2)$$

Designing *negative* pairs in contrastive learning. Several works (Saunshi et al., 2019; Chuang et al., 2020) have come to the awareness of the sampling bias of negative pairs in Equation 2. Specifically, if the negative samples are sampled from $\mathcal{D}$, we will receive with $1/K$ probability a positive sample in a K-class classification task with balanced classes, hence biasing the contrastive loss. To overcome this issue, (Chuang et al., 2020) proposes a *de-biased* constrastive loss to mitigate the sampling bias by explicitly including the class probability prior on the downstream tasks (e.g., with probability 0.1, x_i^- contains a positive example in CIFAR10), and tune the prior τ^+ as a hyperparameter. We denote the loss from (Chuang et al., 2020) as $\mathcal{L}_{\text{Debiased}}$ and the full equation is shown below:

(Debiased loss $\mathcal{L}_{\text{Debiased}}$)

$$\mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D},\\ x^+\sim\mathcal{D}_x^{\text{aug}},\\ v_j\sim\mathcal{D}_x^{\text{aug}},\\ u_i\sim\mathcal{D}_{\backslash x}^{\text{aug}}}}\left[-\log\frac{e^{f(x)^Tf(x^+)}}{e^{f(x)^Tf(x^+)}+Ng_1(x,\{u_i\}^n,\{v_j\}^m)}\right], \quad (3)$$

where the estimator $g_1(x,\{u_i\}^n,\{v_j\}^m)$ is defined by $\max\{\frac{\sum_{i=1}^n e^{f(x)^Tf(u_i)}}{(1-\tau^+)n}-\frac{\tau^+\sum_{j=1}^m e^{f(x)^Tf(v_j)}}{(1-\tau^+)m},e^{-1/t}\}$ and n and m represents the numbers of sampled points in $\mathcal{D}_{\backslash x}^{\text{aug}}$ and $\mathcal{D}_x^{\text{aug}}$ for the re-weighted negative term, τ^+ is the class probability prior, and t is the temperature hyperparameter. Recently, (Robinson et al., 2021) proposes to weigh sample pairs through the cosine distance in the estimator $g_1(x,\{u_i\}^n,\{v_j\}^m)$ based on $\mathcal{L}_{\text{Debiased}}$, and we denote their approach as $\mathcal{L}_{\text{Debiased+HardNeg}}$,

(Debiased+HardNeg loss $\mathcal{L}_{\text{Debiased+HardNeg}}$)

$$\mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D},\\ x^+\sim\mathcal{D}_x^{\text{aug}},\\ v_j\sim\mathcal{D}_x^{\text{aug}},\\ u_i\sim\mathcal{D}_{\backslash x}^{\text{aug}}}}\left[-\log\frac{e^{f(x)^Tf(x^+)}}{e^{f(x)^Tf(x^+)}+Ng_2(x,\{u_i\}^n,\{v_j\}^m)}\right], \quad (4)$$

where the estimator $g_2(x,\{u_i\}^n,\{v_j\}^m)$ is defined by $\max\{\frac{\sum_{i=1}^n \kappa_i^{\beta+1}}{(1-\tau^+)\sum_{i=1}^n\kappa_i^\beta}-\frac{\tau^+\sum_{j=1}^m e^{f(x)^Tf(v_j)}}{(1-\tau^+)m},e^{-1/t}\}$ and $\kappa_i=e^{f(x)^Tf(u_i)}$. A typical choice of n and m are $n=N$ and $m=1$, and the hyperparameter τ^+ in g_2 is exactly the same as that in g_1 whereas the hyperparameter β controls the weighting mechanism. Specifically, when $\tau^+=0$, we denote $\mathcal{L}_{\text{Debiased+HardNeg}}$ as $\mathcal{L}_{\text{HardNeg}}$; when $\beta=0$, Equation (4) degenerates to Equation (3) which is $\mathcal{L}_{\text{Debiased}}$.

Designing *positive* pairs in contrastive learning. Instead of modifying the negative pairs, another direction is to design the positive pairs (Ho & Vasconcelos, 2020; Kim et al., 2020). Specifically, authors of (Ho & Vasconcelos, 2020) define the concept of *adversarial examples* in the regime of representation learning as the positive sample x^{adv} that maximizes $\mathcal{L}_{\text{SimCLR}}$ in Equation (2) within a pre-specified perturbation magnitude ϵ. The resulting loss function is denoted as $\mathcal{L}_{\text{Adv}}$:

(Adversarial loss $\mathcal{L}_{\text{Adv}}$)

$$\mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D},\\ x^+\sim\mathcal{D}_x^{\text{aug}},\\ x_{i_1}^-\sim\mathcal{D}_{\backslash x}^{\text{aug}},\\ x_{i_2}^-\sim\mathcal{D}_{\backslash x}^{\text{adv}}}}\left[-\log\frac{e^{f(x)^Tf(x^+)}}{e^{f(x)^Tf(x^+)}+Ng_0(x,\{x_{i_1}^-\}^N)}\right.$$
$$\left.-\alpha\log\frac{e^{f(x)^Tf(x^{\text{adv}})}}{e^{f(x)^Tf(x^{\text{adv}})}+Ng_0(x,\{x_{i_2}^-\}^N)}\right], \quad (5)$$

where the $\mathcal{D}_{\backslash x}^{\text{adv}}$ is defined by $\cup_{x'\in\mathcal{D}\backslash\{x\}}x'\cup x'^{,\text{adv}}$. Notably, one can adjust the importance of the adversarial term by tuning α in Equation (5).

Adversarial Robustness. Despite neural networks' supremacy in achieving impressive performance, they have been proved vulnerable to human-imperceptible perturbations (Goodfellow et al., 2015; Szegedy et al., 2014; Nguyen et al., 2015; Moosavi-Dezfooli et al., 2016). 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 (Goodfellow et al., 2015), where with a fixed perturbation magnitude ϵ, FGSM finds adversarial perturbation by 1-step gradient descent. Another popular attack method we consider in this paper is PGD (Madry et al., 2018), which assembles the iterative-FGSM (Kurakin et al., 2016) but with different initializations and learning rate constraints.

3. Two New NCA-inspired Contrastive Losses and an Integrated Framework

In this section, we first derive a connection between Neighborhood Component Analysis (NCA) (Goldberger et al., 2004) and the unsupervised contrastive learning loss in Section 3.1. Inspired by our result in Section 3.1, we propose two new NCA-inspired contrastive losses in Section 3.2, which we refer to as **N**eighborhood **a**nalysis **C**ontrastive **l**oss (**NaCl**). To address a lack of robustness in existing contrastive losses, in Section 3.3, we propose a useful framework **IntNaCl** that integrates NaCl and a robustness-promoting loss. A summary of definitions is given as Table S1.

3.1. Bridging from supervised NCA to unsupervised contrastive learning: a new finding

NCA is a supervised learning algorithm concerned with learning a quadratic distance metric with the matrix A such that the performance of nearest neighbour classification is maximized. Notice that the set of neighbors for a data point is a function of transformation A. However, it can remain unchanged as A changes within a certain range. Therefore the leave-one-out classification performance can be a piecewise-constant function of A and hence 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 that point x_j is selected as the nearest neighbor for x_i is given as $p(x_j \mid x_i)$ with

$$p_{ij} := p(x_j \mid x_i) = \frac{e^{-\|Ax_i - Ax_j\|^2}}{\sum_{k \neq i} e^{-\|Ax_i - Ax_k\|^2}}, \; j \neq i. \quad (6)$$

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 in which all points x_j belong to the same class as point x_i. We use M to denote the cardinality of this set. By the definition of c_i, the probability x_i's label is c_i is given as q_i, which is exactly 1[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. Two popular choices for $\ell(\cdot)$ are the total variation distance and the KL divergence. In the seminal paper of (Goldberger et al., 2004), the authors showed both losses give similar results, thus 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 $q_i = 1$. By plugging in the definition of $p_i = \sum_{j|c_j=c_i} p_{ij}$ and Equation 6, the NCA problem becomes

$$\min_A \sum_{i=1}^{n} -\log \left(\sum_{j|c_j=c_i} \frac{e^{-\|Ax_i - Ax_j\|^2}}{\sum_{k \neq i} e^{-\|Ax_i - Ax_k\|^2}} \right). \quad (7)$$

[1]For every data point, p and q are defined differently with their supports being the class index. For every sample x, q_i is the ground truth probability of class labels and p_i is the prediction probability.

With the above formulation, we now show how to establish the connection of NCA to the contrastive learning loss. First, by assuming (a) positive pairs belong to the same class and (b) the transformation Ax is instead parametrized by a general function $\frac{f(x)}{\sqrt{2}} := \frac{h(x)}{\sqrt{2}\|h(x)\|}$, where h is a neural network, we could derive from Equation (7) to Equation (S1) in Appendix A. Next, we show that with some manipulations (details please see Appendix A), below Equation (8) is equivalent to Equation (S1):

$$\min_f \mathbb{E}_{x \sim \mathcal{D}} \left[-\log \left(\frac{\sum_{j=1}^{M} e^{f(x)^T f(x_j^+)}}{\sum_{j=1}^{M} e^{f(x)^T f(x_j^+)} + N g_0(x, \{x_i^-\}^N)} \right) \right]. \quad (8)$$

Notice that Equation (8) is a more general contrastive loss where the contrastive loss $\mathcal{L}_{\text{SimCLR}}$ in (Chen et al., 2020a) is a special case with $M = 1, x^+ \sim \mathcal{D}_x^{\text{aug}}$:

$$\min_f \mathbb{E}_{\substack{x \sim \mathcal{D}, \\ x^+ \sim \mathcal{D}_x^{\text{aug}}, \\ x_i^- \sim \mathcal{D}_{\setminus x}^{\text{aug}}}} \left[-\log \left(\frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)} + N g_0(x, \{x_i^-\}^N)} \right) \right].$$

With the above analysis, two new contrastive losses are proposed based on Equation (8) in the next Section 3.2. As a side note, as the computation of the loss grows quadratically with the size of the dataset, the current method (Chen et al., 2020a) uses mini batches to construct positive/negative pairs in a data batch of size N to estimate the loss.

3.2. Neighborhood analysis Contrastive loss (NaCl)

Based on the connection we have built in Section 3.1, we discover that the reduction from the NCA formulation to $\mathcal{L}_{\text{SimCLR}}$ assumes

1. the expected relative density of positives in the underlying data distribution is $1/N$;
2. the probability q_i induced by encoder network f is 1.

By relaxing the assumptions individually, in this section, we propose two new contrastive losses. Note that the two neighborhood analysis contrastive losses are designed from orthogonal perspectives, hence they are complementary to each other. We use $\mathcal{L}_{\text{NaCl}}$ to denote these two variant losses: $\mathcal{L}_{\text{NCA}}$ and $\mathcal{L}_{\text{MIXNCA}}$.

(I) Relaxing assumption 1: $\mathcal{L}_{\text{NCA}}$. When relating unsupervised SimCLR to supervised NCA, we view two samples in a positive pair as same-class samples. Since in SimCLR, the number of positive pairs $M = 1$, which means that $\{j \mid c_j = c_i\}$ only contains one element. This implies the relative density of positives in the underlying data distribution is $M/N = 1/N$, where N is the data batch size. However, as the expected relative density is task-dependent, it's more reasonable to treat the M/N ratio as a hyperparameter similar to the class probabilities τ^+ introduced by (Chuang et al., 2020). Therefore, we propose the more general contrastive loss $\mathcal{L}_{\text{NCA}}$ which could include more than one element or equivalently $M \neq 1$:

$$(\text{NCA loss } \mathcal{L}_{\text{NCA}}(G = g_0, M))$$

$$\mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D},\\ x_j^+\sim\mathcal{D}_x^{\text{aug}},\\ x_i^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}}} \left[-\log \frac{\sum_{j=1}^{M} e^{f(x)^T f(x_j^+)}}{\sum_{j=1}^{M} e^{f(x)^T f(x_j^+)} + N g_0(x, \{x_i^-\}^N)} \right].$$

We further provide the generalization results as follows: if we let $\mathcal{F}$ be a function class, K be the number of classes, $\mathcal{L}_{\text{Sup}}$ be the cross entropy loss of any downstream K-class classification task, $\widehat{\mathcal{L}}_{\text{NCA}}(g_0, M)$ be the empirical NCA loss, T be the size of the dataset, and $\mathcal{R}_{\mathcal{S}}(\mathcal{F})$ be the empirical Rademacher complexity of $\mathcal{F}$ w.r.t. data sample $\mathcal{S}$, then

Theorem 3.1. *With probability at least $1-\delta$, for any $f \in \mathcal{F}$ and $N \geq K-1$,*

$$\mathcal{L}_{\text{Sup}}(\hat{f}) \leq \mathcal{L}_{\text{NCA}}(g_0, M)(f) + \mathcal{O}\left(\sqrt{\frac{1}{N}} + \frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log \frac{1}{\delta}}{T}}\right),$$

where $\hat{f} = \arg\min_{f\in\mathcal{F}} \widehat{\mathcal{L}}_{\text{NCA}}(g_0, M)(f)$, $\lambda = \frac{1}{M}$, and $B = \log N$.

We can see from the term λ that $\mathcal{L}_{\text{NCA}}(G = g_0, M)$ improves upon $\mathcal{L}_{\text{SimCLR}}$ by using a $M \neq 1$. The result extends to $G = g_1$ and for more details please refer to Appendix B.

(II) Relaxing assumption 2: $\mathcal{L}_{\text{MIXNCA}}$. To reduce the reliance on the downstream prior, a practical relaxation can be made by allowing neighborhood samples to agree with each other with probability. This translates into relaxing 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 match 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

$$(\text{MIXNCA loss } \mathcal{L}_{\text{MIXNCA}}(G = g_0, M, \lambda))$$

$$\mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D},\\ x^+\sim\mathcal{D}_x^{\text{aug}},\\ x_{i_1}^-, x_{i_2 j}^-, x_j^-\sim\mathcal{D}_{\backslash x}^{\text{aug}}}} \left[-\log \frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)} + N g_0(x, \{x_{i_1}^-\}^N)} - \frac{\lambda}{M-1}\sum_{j=1}^{M-1} \log \Omega_j - \frac{1-\lambda}{M-1}\sum_{j=1}^{M-1}\log(1-\Omega_j) \right],$$

where $\Omega_j = \frac{e^{f(x)^T f(\lambda x^+ + (1-\lambda)x_j^-)}}{e^{f(x)^T f(\lambda x^+ + (1-\lambda)x_j^-)} + N g_0(x, \{x_{i_2 j}^-\}_{i_2}^N)}$. Interestingly, the construction of x' herein assembles the mixup (Zhang et al., 2018) philosophy in supervised learning. Recent work (Lee et al., 2021; Verma et al., 2021) have also considered augment the dataset by including synthetic data point and build domain-agnostic contrastive learning strategies, however, their loss is different from this work because they apply mixup on the data points x while we use mixup to produce diverse positive pairs.

3.3. Integrated contrastive learning framework

Building on top of NaCl, we can propose a useful framework **IntNaCl** that not only generalizes existing methods but also achieves good accuracy and robustness simultaneously. Before we introduce IntNaCl, we give an intermediate integrated loss as **IntCl**, which consists of two components – a standard loss and a robustness-promoting loss.

Motivated by $\mathcal{L}_{\text{Adv}}$ (Ho & Vasconcelos, 2020), we consider a robust-promoting loss defined by

$$\mathcal{L}_{\text{Robust}}(G, w) := \mathbb{E}\left[-\log \frac{e^{f(x)^T f(x^{\text{adv}})}}{e^{f(x)^T f(x^{\text{adv}})} + NG(x, \cdot)} w(x) \right],$$

where G can be chose from $\{g_0, g_1, g_2\}$, and $w(x)$ facilitates goal-specific weighting schemes. Note that $w(x)$ can be a general function and $\mathcal{L}_{\text{Adv}}$ (Ho & Vasconcelos, 2020) is a special case when $w(x) = 1$.

Adversarial weighting. Weighting sample loss based on their margins has been proven to be effective in the adversarial training under supervised settings (Zeng et al., 2020). 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 is underdefined in unsupervised settings, we can give our weighting function as the value of the contrastive loss $\hat{w}(x) := -\log \frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)} + NG(x, \cdot)}$. 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. Below, we propose a new integrated framework to involve the robustness term $\mathcal{L}_{\text{Robust}}(G, w)$ which can

Table 1. The relationship between IntNaCl framework and the literature: existing works are special cases of $\mathcal{L}_{\text{IntNaCl}}$

	$\mathcal{L}_{\text{IntNaCl}}$	$\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda)$				α	$\mathcal{L}_{\text{Robust}}(G^2, w))$	
		$\mathcal{L}_{\text{NaCl}}$	G^1	M	λ		G^2	w
Existing Work	$\mathcal{L}_{\text{SimCLR}}$ (Chen et al., 2020a)	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_0	1	-	0	-	-
	$\mathcal{L}_{\text{Debiased}}$ (Chuang et al., 2020)	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_1	1	-	0	-	-
	$\mathcal{L}_{\text{Debiased+HardNeg}}$ (Robinson et al., 2021)	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_2	1	-	0	-	-
	$\mathcal{L}_{\text{Adv}}$ (Ho & Vasconcelos, 2020)	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_0	1	-	1	g_0	1
Our Method	$\mathcal{L}_{\text{IntCl}}$ in Fig. 1	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_2	1	-	1	g_2	$\hat{w}(x)$
	$\mathcal{L}_{\text{IntNaCl}}$ in Fig. 1	$\mathcal{L}_{\text{MIXNCA}}$	g_2	5	0.5	1	g_2	$\hat{w}(x)$
	$\mathcal{L}_{\text{IntNaCl}}$ in Tab. 2	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_0/g_2	1-5	0.5/0.9	0	-	-
	$\mathcal{L}_{\text{IntNaCl}}$ in Tab. 3	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_2	1-5	0.5/0.7/0.9	1	g_2	$\hat{w}(x)$
	$\mathcal{L}_{\text{IntNaCl}}$ in Fig. S2	$\mathcal{L}_{\text{MIXNCA}}$	g_0/g_2	1-5	0.5-0.9	0	-	-
	$\mathcal{L}_{\text{IntNaCl}}$ in Tab. 4	$\mathcal{L}_{\text{NCA}}/\mathcal{L}_{\text{MIXNCA}}$	g_0/g_2	1/2/5	0.5/0.9	0/1	-/g_2	-/$\hat{w}(x)$/1

greatly help on promoting robustness in contrastive learning. In particular, we show that many existing contrastive learning losses are special cases of our proposed framework.

IntCl. For IntCl, the standard loss can be existing contrastive learning losses (Chen et al., 2020a; Chuang et al., 2020; Robinson et al., 2021), which correspond to a form of

(IntCL loss $\mathcal{L}_{\text{IntCL}}$)

$$\mathbb{E}\left[-\log \frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)} + NG^1(x,\cdot)}\right] + \alpha\mathcal{L}_{\text{Robust}}(G^2, w),$$

with G^1 and G^2 being g_0, g_1, and g_2. Unless otherwise specified, we use the adversarial weighting scheme introduced above throughout our experiments. Notice that $\mathcal{L}_{\text{IntCL}}$ reduces to $\mathcal{L}_{\text{Adv}}$ when $G^1 = G^2 = g_0$ and $w(x) \equiv 1$.

IntNaCl. To design a generic loss that accounts for robust accuracy while keeping clean accuracy, we utilize $\mathcal{L}_{\text{NaCl}}$ developed in Section 3.2 to strength the standard loss in $\mathcal{L}_{\text{IntCL}}$. We call this ultimate framework **Int**egrated **N**eighborhood **a**nalysis **C**ontrastive **l**oss (**IntNaCl**), which is given by

$$\mathcal{L}_{\text{IntNaCl}} := \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) + \alpha\mathcal{L}_{\text{Robust}}(G^2, w), \quad (9)$$

where $\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda)$ can be chose from $\{\mathcal{L}_{\text{NCA}}(G^1, M), \mathcal{L}_{\text{MIXNCA}}(G^1, M, \lambda)\}$. We remark that as $\mathcal{L}_{\text{NCA}}$ and $\mathcal{L}_{\text{MIXNCA}}$ all reduce to one same form when $M = 1$, the $\mathcal{L}_{\text{IntNaCl}}$ under $M = 1$ is exactly $\mathcal{L}_{\text{IntCl}}$. This general framework includes many of the existing works as special cases and we summarize these relationships in Table 1.

4. Experimental Results

Implementation details. All the proposed methods are implemented based on open source repositories provided in the literature (Chen et al., 2020a; Ho & Vasconcelos, 2020; Robinson et al., 2021). Five benchmarking contrastive losses are considered as baselines that include: $\mathcal{L}_{\text{SimCLR}}$ (Chen et al., 2020a), $\mathcal{L}_{\text{Debiased}}$ (Chuang et al., 2020), $\mathcal{L}_{\text{Debiased+HardNeg}}$ (Robinson et al., 2021), $\mathcal{L}_{\text{Adv}}$ (Ho & Vasconcelos, 2020) (i.e. Equation (2), Equation (3), Equation (4), Equation (5)). We train representations on resnet18 and include MLP projection heads (Chen et al., 2020a). A batch size of 256 is used for all CIFAR (Krizhevsky et al., 2009) experiments and a batch size of 128 is used for all tinyImagenet experiments. 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. We implement the proposed framework using PyTorch to enable the use of an NVIDIA GeForce RTX 2080 Super GPU and four NVIDIA Tesla V100 GPUs.

Evaluation protocol. We follow the standard evaluation protocal to report three major properties of representation learning methods: standard discriminative power, transferability, and adversarial robustness. To evaluate the standard discriminative power, we train representation networks on CIFAR100/tinyImagenet, freeze the network, and fine-tune a fully-connected layer that maps representations to outputs on CIFAR100/tinyImagenet, which is consistent with the standard linear evaluation protocol in the literature (Chen et al., 2020a; Chuang et al., 2020; Grill et al., 2020; Ho & Vasconcelos, 2020; Khosla et al., 2020; Tian et al., 2020; Robinson et al., 2021; Saunshi et al., 2019; Kim et al., 2020; HaoChen et al., 2021). To evaluate the transferability, we use the representation networks trained on CIFAR100, and only fine-tune a fully-connected layer that maps representations to outputs on CIFAR10. All the adversarial robustness evaluations are based on the implementation provided by (Wong et al., 2020). We supplement more FGSM and PGD attack results in the appendix.

Experiment outline. Since the performance of the integrated method $\mathcal{L}_{\text{IntNaCl}}$ is attributed to multiple components in the formulation (Equation 9), we do ablation studies in the following sections to study their effectiveness individually. In Section 4.1, we evaluate the effect of $\mathcal{L}_{\text{NaCl}}$; in Section 4.2, we evaluate the effect of $\mathcal{L}_{\text{Robust}}$; in Section 4.3, we evaluate the effect of M, λ, and w.

Table 2. Performance comparisons of $\mathcal{L}_{\text{NaCl}}$ ($M \neq 1$) and i) *Left*: SimCLR (Chuang et al., 2020) ($M = 1, G^1 = g_0$) and ii) *Right*: Debised+HardNeg (Robinson et al., 2021) ($M = 1, G^1 = g_2$) when $\alpha = 0$. The best accuracy (%) within each loss type is in boldface (larger is better).

M	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_0, M)$				$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, M)$			
	CIFAR100	CIFAR100 Adv.	CIFAR10	CIFAR10 Adv.	CIFAR100	CIFAR100 Adv.	CIFAR10	CIFAR10 Adv.
1	53.69±0.25	25.17±0.55	76.34±0.28	43.50±0.41	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	55.72±0.15	27.04±0.45	77.40±0.14	44.58±0.41	57.87±0.15	32.50±0.48	77.43±0.11	48.14±0.31
3	56.67±0.12	**28.41±0.24**	77.53±0.24	**45.21±0.89**	58.42±0.23	**33.19±0.60**	77.41±0.17	48.09±0.93
4	57.09±0.26	28.20±0.81	77.75±0.22	45.13±0.44	**58.86±0.18**	32.65±1.07	77.46±0.29	**48.43±0.94**
5	**57.32±0.17**	28.33±0.59	**77.93±0.40**	44.46±0.53	58.81±0.21	32.86±0.47	**77.58±0.23**	48.30±0.39
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_0, M, 0.9)$				$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$			
1	53.69±0.25	25.17±0.55	76.34±0.28	43.50±0.41	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	56.20±0.33	30.95±0.36	76.96±0.15	**48.85±0.75**	59.41±0.19	**32.22±0.35**	79.36±0.65	48.86±0.34
3	56.41±0.13	**30.98±0.90**	77.10±0.21	48.76±0.63	59.81±0.25	32.04±0.67	79.41±0.17	48.91±0.81
4	56.00±0.42	29.90±0.63	**77.11±0.40**	48.16±0.40	59.75±0.33	32.03±0.34	79.42±0.18	**49.05±0.71**
5	**56.63±0.31**	30.58±0.52	77.04±0.19	47.96±0.46	**59.85±0.30**	32.06±0.72	**79.45±0.20**	48.32±0.70

Table 3. Performance comparisons of $\mathcal{L}_{\text{IntNaCl}}$ ($M \neq 1$) and $\mathcal{L}_{\text{IntCL}}$ ($M = 1$) when $\alpha = 1, G^1 = G^2 = g_2, w = \hat{w}(x)$. The best accuracy (%) within each loss type is in boldface (larger is better).

M	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, M)$				$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$			
	CIFAR100	CIFAR100 Adv.	CIFAR10	CIFAR10 Adv.	CIFAR100	CIFAR100 Adv.	CIFAR10	CIFAR10 Adv.
1	56.22±0.15	40.05±0.67	76.39±0.10	**59.33±0.94**	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	58.97±0.19	40.25±0.52	78.61±0.20	58.41±0.59
3	57.13±0.26	40.53±0.29	**76.67±0.22**	58.47±0.31	59.26±0.18	40.96±0.58	**78.83±0.22**	59.20±1.25
4	57.06±0.19	40.85±0.31	76.34±0.22	58.91±0.62	59.32±0.21	40.82±0.54	78.83±0.27	59.03±0.52
5	**57.46±0.04**	**41.00±0.86**	76.60±0.37	57.98±0.47	**59.43±0.23**	**41.01±0.34**	78.80±0.21	**59.51±0.93**
	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.7)$				$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.9)$			
1	56.22±0.15	40.05±0.67	76.39±0.10	59.33±0.94	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	56.54±0.33	40.85±0.13	76.81±0.22	60.40±0.46
3	58.23±0.18	40.94±0.75	77.91±0.25	**59.57±0.81**	56.69±0.11	41.23±0.66	**76.98±0.22**	60.13±0.56
4	58.20±0.25	40.95±0.45	77.89±0.20	59.49±0.49	56.43±0.26	**41.56±0.56**	76.97±0.20	**61.21±0.49**
5	**58.37±0.14**	**41.15±0.48**	**78.27±0.26**	59.17±0.94	**56.86±0.11**	41.09±0.31	76.91±0.21	60.09±0.39

Table 4. Performance comparisons of $\mathcal{L}_{\text{NaCl}}$ and $\mathcal{L}_{\text{IntNaCl}}$ with baselines on TinyImagenet. The best accuracy (%) within each loss type is in boldface (larger is better).

$\alpha = 0$	$\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_0, M)$		$\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, M)$	
M	TinyImagenet	TinyImagenet Adv.	TinyImagenet	TinyImagenet Adv.
1	39.66±0.15	24.80±0.07	41.26±0.14	27.34±0.77
2	**40.71±0.26**	**26.29±0.51**	**41.99±0.23**	**28.14±0.13**
	$\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_0, M, 0.9)$		$\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$	
1	39.66±0.15	24.80±0.07	41.26±0.14	27.34±0.77
2	**40.23±0.37**	**26.47±0.24**	**43.91±0.20**	**28.29±0.33**
$\alpha = 1$	$\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$ $\mathcal{L}_{\text{Robust}}(G^2, w)) = \mathcal{L}_{\text{Robust}}(g_2, \hat{w}(x)))$		$\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$ $\mathcal{L}_{\text{Robust}}(G^2, w)) = \mathcal{L}_{\text{Robust}}(g_2, 1))$	
1	42.56±0.13	31.18±0.51	42.24±0.14	31.55±0.38
2	44.69±0.20	**32.65±0.52**	44.37±0.08	32.20±0.23
5	**45.31±0.22**	32.43±0.33	**44.77±0.11**	**32.47±0.42**

4.1. The effect of $\mathcal{L}_{\text{NaCl}}$

By evaluating the effect of $\mathcal{L}_{\text{NaCl}}$, we want to evaluate the performance difference of our framework $\mathcal{L}_{\text{IntNaCl}}$ when $M \geq 1$ and $M = 1$. In order to see that, we consider 2 cases: (1) set $\alpha = 0$ in Equation (9) and compare $\mathcal{L}_{\text{NaCl}}(G^1, M \neq 1, \lambda)$ with existing work $\mathcal{L}_{\text{NaCl}}(G^1, M = 1, \lambda)$, or (2) set $\alpha = 1$ and compare $\mathcal{L}_{\text{IntNaCl}}$ and $\mathcal{L}_{\text{IntCl}}$.

Case (1) $\alpha = 0$**.** In Table 2, after setting $\alpha = 0$, we experiment with $G^1 = g_0, g_2$. By referring to Table 1, our baseline becomes exactly SimCLR (Chen et al., 2020a) when $G^1 = g_0$, and becomes Debiased+HardNeg (Robinson et al., 2021) when $G^1 = g_2$. From Table 2, one can see that when $M \neq 1$, $\mathcal{L}_{\text{NCA}}$ and $\mathcal{L}_{\text{MIXNCA}}$ can both improve upon the baselines($M = 1$) in all metrics (standard/robust/transfer accuracy). When $G^1 = g_0$, $\mathcal{L}_{\text{NCA}}$'s improvement over SimCLR also exemplifies our Theorem 3.1. Due to page limits, we only select one λ when $\mathcal{L}_{\text{NaCl}} = \mathcal{L}_{\text{MIXNCA}}$ and report results together with the results of $\mathcal{L}_{\text{NaCl}} = \mathcal{L}_{\text{NCA}}$. Full tables can be found in the appendix D. We further verify the performance on TinyImagent and give results in Table 4. Notice that now when $G^1 = g_0$, we are using a batch size

Table 5. Combining $\mathcal{L}_{\text{NCA}}(g_2, 5)$ and $\mathcal{L}_{\text{MIXNCA}}(g_2, 5, 0.5)$.

Accuracy (%)	CIFAR100	CIFAR100 Adv.	CIFAR10	CIFAR10 Adv.
$\mathcal{L}_{\text{NCA}}$	58.81±0.21	32.86±0.47	77.58±0.23	48.30±0.39
$\mathcal{L}_{\text{MIXNCA}}$	59.85±0.30	32.06±0.72	79.45±0.20	48.32±0.70
Combined	59.66±0.14	**33.64±0.31**	78.94±0.07	**51.19±0.44**

of $N = 128$ for 200-class TinyImagent task. Therefore, the requirement of $N \geq K - 1$ in Theorem 3.1 is not fulfilled. However, we can still see improvements when going from $M = 1$ to $M = 2$. Additionally, we combine $\mathcal{L}_{\text{NCA}}$ and $\mathcal{L}_{\text{MIXNCA}}$ in training and give their results in Table 5. We see that the robustness performance can be further boosted by 1-3% with the combined loss while keeping similar standard accuracy to $\mathcal{L}_{\text{MIXNCA}}$.

Case (2) $\alpha = 1$**.** In Table 3, after setting $\alpha = 1$, we experiment with $G^1 = G^2 = g_2$ since g_2 generally yields better performance in Table 2. When $\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, \lambda)$, we give the results for $\lambda = 0.5, 0.7, 0.9$ to show an interesting effect: while $\mathcal{L}_{\text{MIXNCA}}(g_2, M, \lambda = 0.5)$ benefits a lot going from $M = 1$ to $M = 5$ (standard accuracy increases from 56.22% to 59.43%), the improvement is comparatively smaller with $\mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.9)$ (standard accuracy increases from 56.22% to 56.86%). In Figure 1, we plot the robust accuracy defined under FGSM attacks (Goodfellow et al., 2015) along the y-axis. Ideally, one desires a representation network that pushes the performance to the upper-right corner in the 2D accuracy grid (standard-robust accuracy plot). We highlight the results of $\mathcal{L}_{\text{IntNaCl}}$ and $\mathcal{L}_{\text{IntCL}}$ in circles, through which we see that while $\mathcal{L}_{\text{IntCL}}$ can already train representations that are decently robust without sacrificing the standard accuracy on CIFAR100, the standard accuracy on CIFAR10 is inferior to some baselines (HardNeg and Debiased+HardNeg). Comparatively, $\mathcal{L}_{\text{IntNaCl}}$ demonstrates high transfer standard accuracy and wins over the baselines by a large margin on both datasets, proving the ability of learning representation networks that also transfer robustness property. For TinyImagent, we only show the results when $\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$ since g_2 generally achieves higher accuracy and combines well with $\mathcal{L}_{\text{MIXNCA}}$. Importantly, with the help of $\mathcal{L}_{\text{NaCl}}$ module, the performance can be boosted from 42.56% to 45.31% while maintaining good robust accuracy 32.43%.

4.2. The effect of $\mathcal{L}_{\text{Robust}}$

By evaluating the effect of $\mathcal{L}_{\text{Robust}}$, we want to see the performance difference of our framework $\mathcal{L}_{\text{IntNaCl}}$ when $\alpha \neq 0$ and $\alpha = 0$. Therefore, we consider 2 cases: (1) set $M = 1$ in Equation (9) and compare $\mathcal{L}_{\text{IntCl}}$ with existing work $\mathcal{L}_{\text{NaCl}}(G^1, M = 1, \lambda)$, or (2) set $M \neq 1$ and compare $\mathcal{L}_{\text{IntNaCl}}$ and $\mathcal{L}_{\text{NaCl}}(G^1, M \neq 1, \lambda)$.

Case (1) $M = 1$**.** Notice that $\mathcal{L}_{\text{IntCL}}$ differs from standard contrastive losses by including the term $\mathcal{L}_{\text{Robust}}$. Therefore, one can easily evaluate the effect of $\mathcal{L}_{\text{Robust}}$ by inspecting the performance difference between $\mathcal{L}_{\text{IntCl}}$ and the baselines in Figure 1. Specifically, we let $G^1 = g_2$ for $\mathcal{L}_{\text{IntCl}}$ in Figure 1, hence a direct baseline is Debiased+HardNeg. By adding a robustness-promoting term, the robust accuracy can be boosted from 31.03% to 40.05% and transfer robust accuracy from 48.38% to 59.33%, which is a significant improvement.

Case (2) $M \neq 1$**.** The effect of $\mathcal{L}_{\text{Robust}}$ is also demonstrated through the robust accuracy "jump" from Table 2 to Table 3. For example, we point out that in Table 2, $\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, 3)$ gives the maximum robust accuracy of 33.19%, while the robust accuracy obtained with the same $\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, 3)$ and additional $\mathcal{L}_{\text{Robust}}$ increases to 40.53% in Table 3. The robust accuracy boost on TinyImagent with the help of $\mathcal{L}_{\text{Robust}}$ is also visible: when $\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, 2, 0.5)$, the robust accuracy increases from 28.29% to 32.65%.

4.3. The effect of M, λ, and $w(x)$

To evaluate the effect of M, we can see from Table 2 and Table 3 that the performance is generally increasing as M increases. However, this effect seems to be less visible for robust accuracy and transfer robust accuracy. In practice, $M = 5$ does not require exactly 5 times training time since the number of training parameter remains the same. In our experiment, we observe that $M = 5$ requires roughly 3 times training time compared with the baseline $M = 1$. To evaluate the effect of λ, we include in Figure S2 the standard and robust accuracy on CIFAR100 and CIFAR10 as functions of λ. Intriguingly, we see that the accuracy curves mainly show trends of increasing in Figure S2(a). Comparatively, the standard accuracy on CIFAR100 and CIFAR10 shows trends of decreasing in Figure S2(b). One possible explanation is by the original baselines' room for improvement. Since Debiased+HardNeg is a much stronger baseline than SimCLR, it is closer to the robustness-accuracy trade-off. However, we note that the overall performance of NaCl on Debiased+HardNeg is still better than NaCl on SimCLR regardless of the robustness-accuracy trade-off. In the last row of Table 4, we list the results when $\mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$ but different $\mathcal{L}_{\text{Robust}}(G^2, w))$. Specifically, on the left we show the case when $w = \hat{w}(x)$ and on the right we show the case when $w = 1$. One can then see that by using a goal-specific weighting scheme, the performance can be further boosted.

5. Conclusion

In this paper, we discover the relationship between contrastive loss and Neighborhood Component Analysis (NCA), which motivates us to generalize the existing contrastive loss to a set of Neighborhood analysis Contrastive losses (NaCl). We further propose a generic and integrated contrastive learning framework (IntNaCl) based on NaCl, which learns representations that score high in both standard accuracy and adversarial accuracy in downstream tasks. With the integrated framework, we can boost the standard accuracy by 6% and the robust accuracy by 17%.

Acknowledgements

This work was partially supported by the MIT-IBM Watson AI Lab and by the National Science Foundation.

References

Bojanowski, P. and Joulin, A. Unsupervised learning by predicting noise. In *ICML*, pp. 517–526. PMLR, 2017.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607, Virtual, 13–18 Jul 2020a. PMLR.

Chen, X., Fan, H., Girshick, R. B., and He, K. Improved baselines with momentum contrastive learning. *CoRR*, abs/2003.04297, 2020b. URL https://arxiv.org/abs/2003.04297.

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

Dosovitskiy, A., Fischer, P., Springenberg, J. T., Riedmiller, M., and Brox, T. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2015.

Goldberger, J., Hinton, G. E., Roweis, S., and Salakhutdinov, R. R. Neighbourhood components analysis. *NeurIPS*, 17: 513–520, 2004.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

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

Gutmann, M. and Hyvärinen, A. 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.

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

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

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

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

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

Kim, M., Tack, J., and Hwang, S. J. Adversarial self-supervised contrastive learning. *arXiv preprint arXiv:2006.07589*, 2020.

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

Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial machine learning at scale. In *ICLR*, 2016.

Lee, K., Zhu, Y., Sohn, K., Li, C.-L., Shin, J., and Lee, H. i-mix: A domain-agnostic strategy for contrastive representation learning. In *ICLR*, 2021.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Mnih, A. and Teh, Y. W. A fast and simple algorithm for training neural probabilistic language models. In *ICML*, 2012.

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

Nguyen, A., Yosinski, J., and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 2015.

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

Robinson, J. D., Chuang, C.-Y., Sra, S., and Jegelka, S. Contrastive learning with hard negative samples. In *ICLR*, 2021.

Saunshi, N., Plevrakis, O., Arora, S., Khodak, M., and Khandeparkar, H. A theoretical analysis of contrastive unsupervised representation learning. In *ICML*, pp. 5628–5637, 2019.

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

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., and Isola, P. What makes for good views for contrastive learning? In *NeurIPS*, volume 33, pp. 6827–6839, 2020.

Verma, V., Luong, T., Kawaguchi, K., Pham, H., and Le, Q. Towards domain-agnostic contrastive learning. In *International Conference on Machine Learning*, pp. 10530–10541. PMLR, 2021.

Wang, X. and Gupta, A. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802, 2015.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020.

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

Ye, M., Zhang, X., Yuen, P. C., and Chang, S.-F. Unsupervised embedding learning via invariant and spreading instance feature. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6210–6219, 2019.

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

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

A. Derivation from Equation (7) to (8)

Since a generalization of contrastive learning loss in Equation (2) can be given by assuming (a) positive pairs belong to the same class and (b) the transformation Ax is instead parametrized by a general function $\frac{f(x)}{\sqrt{2}} := \frac{h(x)}{\sqrt{2}\|h(x)\|}$, where h is a neural network, Equation (7) becomes Equation (S1):

$$\min_f \sum_{i=1}^{n} -\log\left(\sum_{j=1}^{M} \frac{e^{-\frac{1}{2}\left\|f(x_i)-f(x_{ij}^{+})\right\|^2}}{\sum_{k\neq i} e^{-\frac{1}{2}\|f(x_i)-f(x_k)\|^2}}\right). \tag{S1}$$

Then we can prove

$$\begin{aligned}
&\arg\min_f \sum_{i=1}^{n} -\log\left(\sum_{j=1}^{M} \frac{e^{-\frac{1}{2}\left\|f(x_i)-f(x_{ij}^{+})\right\|^2}}{\sum_{k\neq i} e^{-\frac{1}{2}\|f(x_i)-f(x_k)\|^2}}\right)\\
&=\arg\min_f \sum_{i=1}^{n} -\log\left(\sum_{j=1}^{M} \frac{e^{f(x_i)^T f(x_{ij}^{+})-\frac{1}{2}\|f(x_i)\|^2-\frac{1}{2}\left\|f(x_{ij}^{+})\right\|^2}}{\sum_{k\neq i} e^{f(x_i)^T f(x_k)-\frac{1}{2}\|f(x_i)\|^2-\frac{1}{2}\|f(x_k)\|^2}}\right) &\text{(S2)}\\
&=\arg\min_f \sum_{i=1}^{n} -\log\left(\sum_{j=1}^{M} \frac{e^{f(x_i)^T f(x_{ij}^{+})-1}}{\sum_{k\neq i} e^{f(x_i)^T f(x_k)-1}}\right) &\text{(S3)}\\
&=\arg\min_f \sum_{i=1}^{n} -\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x_i)^T f(x_{ij}^{+})}}{\sum_{k\neq i} e^{f(x_i)^T f(x_k)}}\right)\\
&=\arg\min_f \sum_{i=1}^{n} -\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x_i)^T f(x_{ij}^{+})}}{\sum_{k\neq i, x_k\in\{x_{ij}^{+}\}} e^{f(x_i)^T f(x_k)} + \sum_{k\neq i, x_k\notin\{x_{ij}^{+}\}} e^{f(x_i)^T f(x_k)}}\right) &\text{(S4)}\\
&=\arg\min_f \mathbb{E}_{x\sim\mathcal{D}}\left[-\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^{+})}}{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^{+})} + \sum\limits_{i=1}^{N} e^{f(x)^T f(x_i^{-})}}\right)\right] &\text{(S5)}\\
&=\arg\min_f \mathbb{E}_{x\sim\mathcal{D}}\left[-\log\left(\frac{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^{+})}}{\sum\limits_{j=1}^{M} e^{f(x)^T f(x_j^{+})} + N g_0(x,\{x_i^{-}\}^N)}\right)\right],
\end{aligned}$$

where we go from Equation (S2) to Equation (S3) based on the fact that $\|f(x)\| = 1$, and from Equation (S4) to Equation (S5) assuming that set $\{x_k : k \neq i\} = \{x_j^{+} : 1 \leq j \leq M\} \cup \{x_i^{-} : 1 \leq i \leq N\}$.

B. Generalization Bounds

We extend the theorems from (Chuang et al., 2020) to get results for $\mathcal{L}_{\text{NCA}}$. The results we have here apply to $G = g_0$ and g_1. The case when $G = g_2$, $\mathcal{L}_{\text{MIXNCA}}$, and $\mathcal{L}_{\text{IntNaCl}}$ are left as future work.

B.1. Bridging the empirical estimator and asymptotic objective

We introduce an intermediate unbiased loss in order to extend our results. Let $h(x, y) = e^{f(x)^\top f(y)}$, then the unbiased loss with multiple positive pairs is given as

$$\widetilde{L}^{M,N}_{\text{Unbiased}}(f) = \mathbb{E}_{\substack{x\sim p \\ x_i^+\sim p_x^+}} \left[\log \frac{\sum_{i=1}^M h(x, x_i^+)}{\sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot \mathbb{E}_{x^-\sim p_x^-} h(x, x^-)} \right]$$

Then we can define a debiased loss by

$$L^{M,N,n,m}_{\text{Debiased}}(f) = \mathbb{E}_{\substack{x\sim p \\ x_i^+\sim p_x^+ \\ u_i\sim p; v_i\sim p_x^+}} \left[\log \frac{\sum_{i=1}^M h(x, x_i^+)}{\sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m)} \right].$$

Theorem B.1. *For any embedding f and finite N and M, we have*

$$\left| \widetilde{L}^{M,N}_{\text{Unbiased}}(f) - L^{M,N,n,m}_{\text{Debiased}}(f) \right| \leq \frac{e^{3/2}}{\tau^-}\sqrt{\frac{\pi}{2n}} + \frac{e^{3/2}\tau^+}{\tau^-}\sqrt{\frac{\pi}{2m}}.$$

The proof of B.1 is the same as the proof of Theorem 3 in (Chuang et al., 2020) with the help of the following slightly modified version of Lemma A.2 in (Chuang et al., 2020). Now if we let

$$\Delta = \left| -\log \frac{\sum_{i=1}^M h(x, x_i^+)}{\sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m)} + \log \frac{\sum_{i=1}^M h(x, x_i^+)}{\sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot \mathbb{E}_{x^-\sim p_x^-} h(x, x^-)} \right|,$$

where $h(x, \bar{x}) = \exp^{f(x)^\top f(\bar{x})}$, then one has the following lemma:

Lemma B.2. *Let x and x^+ in $\mathcal{X}$ be fixed. Further, let $\{u_i\}_{i=1}^n$ and $\{v_i\}_{i=1}^m$ be collections of i.i.d. random variables sampled from p and p_x^+ respectively. Then for all $\varepsilon > 0$,*

$$\mathbb{P}(\Delta \geq \varepsilon) \leq 2\exp\left(-\frac{n\varepsilon^2(\tau^-)^2}{2e^3}\right) + 2\exp\left(-\frac{m\varepsilon^2(\tau^-/\tau^+)^2}{2e^3}\right).$$

Proof of Lemma B.2. We first decompose the probability as

$$\begin{aligned}
&\mathbb{P}(\left| -\log \frac{\sum_{i=1}^M h(x, x_i^+)}{\sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m)} + \log \frac{\sum_{i=1}^M h(x, x_i^+)}{\sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot \mathbb{E}_{x^-\sim p_x^-} h(x, x^-)} \right| \geq \varepsilon) \\
&= \mathbb{P}(\left| \log\left\{ \sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m) \right\} - \log\left\{ \sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot \mathbb{E}_{x^-\sim p_x^-} h(x, x^-) \right\} \right| \geq \varepsilon) \\
&= \mathbb{P}(\log\left\{ \sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m) \right\} - \log\left\{ \sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot \mathbb{E}_{x^-\sim p_x^-} h(x, x^-) \right\} \geq \varepsilon) \\
&+ \mathbb{P}(-\log\left\{ \sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m) \right\} + \log\left\{ \sum_{i=1}^M h(x, x_i^+) + M \cdot N \cdot \mathbb{E}_{x^-\sim p_x^-} h(x, x^-) \right\} \geq \varepsilon)
\end{aligned}$$

where the final equality holds simply because $|X| \geq \varepsilon$ if and only if $X \geq \varepsilon$ or $-X \geq \varepsilon$. The first term can be bounded as

$$\begin{aligned}
&\mathbb{P}(\log\big\{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m)\big\} - \log\big\{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)\big\} \geq \varepsilon) \\
&= \mathbb{P}(\log \frac{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m)}{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)} \geq \varepsilon) \\
&\leq \mathbb{P}(\frac{M\cdot N\cdot G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m) - M\cdot N\cdot \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)}{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)} \geq \varepsilon) \\
&= \mathbb{P}(G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m) - \mathbb{E}_{x^-\sim p_x^-} h(x,x^-) \geq \varepsilon\Big\{\frac{1}{M\cdot N}\sum_{i=1}^{M} h(x,x_i^+) + \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)\Big\}) \\
&\leq \mathbb{P}(G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m) - \mathbb{E}_{x^-\sim p_x^-} h(x,x^-) \geq \varepsilon e^{-1}). \qquad (10)
\end{aligned}$$

The first inequality follows by applying the fact that $\log x \leq x-1$ for $x>0$. The second inequality holds since $\frac{1}{M\cdot N\cdot}\sum_{i=1}^{M} h(x,x_i^+) + \mathbb{E}_{x^-\sim p_x^-} h(x,x^-) \geq e^{-1}$. Next, we move on to bounding the second term, which proceeds similarly, using the same two bounds.

$$\begin{aligned}
&\mathbb{P}\Big\{-\log\big(\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m)\big\} + \log\big\{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)\big\} \geq \varepsilon) \\
&= \mathbb{P}(\log \frac{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)}{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m)} \geq \varepsilon) \\
&\leq \mathbb{P}(\frac{M\cdot N\cdot \mathbb{E}_{x^-\sim p_x^-} h(x,x^-) - M\cdot N\cdot G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m)}{\sum_{i=1}^{M} h(x,x_i^+) + M\cdot N\cdot G(x,\{u_i\}_{i=1}^N,\{v_i\}_{i=1}^M)} \geq \varepsilon) \\
&= \mathbb{P}(\mathbb{E}_{x^-\sim p_x^-} h(x,x^-) - G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m) \geq \varepsilon\Big\{\frac{1}{M\cdot N}\sum_{i=1}^{M} h(x,x_i^+) + G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m)\Big\}) \\
&\leq \mathbb{P}(\mathbb{E}_{x^-\sim p_x^-} h(x,x^-) - G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m) \geq \varepsilon e^{-1}). \qquad (11)
\end{aligned}$$

Combining equation (10) and equation (11), we have

$$\mathbb{P}(\Delta \geq \varepsilon) \leq \mathbb{P}(\big|G(x,\{u_i\}_{i=1}^n,\{v_i\}_{i=1}^m) - \mathbb{E}_{x^-\sim p_x^-} h(x,x^-)\big| \geq \varepsilon e^{-1}).$$

Lastly, one needs to bound the right hand tail probability. This part of the proof remains exactly the same as in (Chuang et al., 2020) and is therefore omitted.

□

B.2. Bridging the asymptotic objective and supervised loss

Lemma B.3. *For any embedding f, whenever $N \geq K-1$ we have*

$$L_{\text{Sup}}(f) \leq L^{\mu}_{\text{Sup}}(f) \leq \widetilde{L}^{M,N}_{\text{Unbiased}}(f).$$

Proof. We first show that $N = K-1$ gives the smallest loss:

$$\begin{aligned}
\widetilde{L}^{M,N}_{\text{Unbiased}}(f) &= \mathbb{E}_{\substack{x\sim p \\ x_i^+\sim p_x^+}}\left[-\log \frac{\sum_{i=1}^{M} e^{f(x)^T f(x_i^+)}}{\sum_{i=1}^{M} e^{f(x)^T f(x_i^+)} + M\cdot N\mathbb{E}_{x^-\sim p_x^-} e^{f(x)^T f(x^-)}}\right] \\
&\geq \mathbb{E}_{\substack{x\sim p \\ x_i^+\sim p_x^+}}\left[-\log \frac{\sum_{i=1}^{M} e^{f(x)^T f(x_i^+)}}{\sum_{i=1}^{M} e^{f(x)^T f(x_i^+)} + M\cdot (K-1)\mathbb{E}_{x^-\sim p_x^-} e^{f(x)^T f(x^-)}}\right] \\
&= L^{M,K-1}_{\text{Unbiased}}(f)
\end{aligned}$$

To show that $L^{M,K-1}_{\text{Unbiased}}(f)$ is an upper bound on the supervised loss $L_{\text{sup}}(f)$, we additionally introduce a task specific class distribution $\rho_{\mathcal{T}}$ which is a uniform distribution over all the possible K-way classification tasks with classes in $\mathcal{C}$. That is, we consider all the possible task with K distinct classes $\{c_1, \ldots, c_K\} \subseteq \mathcal{C}$.

$$
\begin{aligned}
& L^{M,K-1}_{\text{Unbiased}}(f) \\
&= \mathbb{E}_{\substack{x\sim p \\ x_i^+ \sim p_x^+}} \left[-\log \frac{\sum_{i=1}^M e^{f(x)^T f(x_i^+)}}{\sum_{i=1}^M e^{f(x)^T f(x_i^+)} + M\cdot(K-1)\mathbb{E}_{x^-\sim p_x^-} e^{f(x)^T f(x^-)}} \right] \\
&= \mathbb{E}_{\mathcal{T}\sim\mathcal{D}} \mathbb{E}_{\substack{c\sim\rho_{\mathcal{T}};x\sim p(\cdot|c) \\ x_i^+\sim p(\cdot|c)}} \left[-\log \frac{\sum_{i=1}^M e^{f(x)^T f(x_i^+)}}{\sum_{i=1}^M e^{f(x)^T f(x_i^+)} + M\cdot(K-1)\mathbb{E}_{\mathcal{T}\sim\mathcal{D}}\mathbb{E}_{\rho_{\mathcal{T}}(c^- \sim | c^- \neq h(x))}\mathbb{E}_{x^-\sim p(\cdot|c^-)} e^{f(x)^T f(x^-)}} \right] \\
&\geq \mathbb{E}_{\mathcal{T}\sim\mathcal{D}} \mathbb{E}_{c\sim\rho_{\mathcal{T}};x\sim p(\cdot|c)} \left[-\log \frac{\sum_{i=1}^M e^{f(x)^T \mathbb{E}_{x_i^+\sim p(\cdot|c)} f(x_i^+)}}{\sum_{i=1}^M e^{f(x)^T \mathbb{E}_{x_i^+\sim p(\cdot|c)} f(x_i^+)} + M\cdot(K-1)\mathbb{E}_{\mathcal{T}\sim\mathcal{D}}\mathbb{E}_{\rho_{\mathcal{T}}(c^- | c^- \neq h(x))}\mathbb{E}_{x^-\sim p(\cdot|c^-)} e^{f(x)^T f(x^-)}} \right] \\
&\geq \mathbb{E}_{\mathcal{T}\sim\mathcal{D}} \mathbb{E}_{c\sim\rho_{\mathcal{T}};x\sim p(\cdot|c)} \left[-\log \frac{\sum_{i=1}^M e^{f(x)^T \mathbb{E}_{x_i^+\sim p(\cdot|c)} f(x_i^+)}}{\sum_{i=1}^M e^{f(x)^T \mathbb{E}_{x_i^+\sim p(\cdot|c)} f(x_i^+)} + M\cdot(K-1)\mathbb{E}_{\rho_{\mathcal{T}}(c^- | c^- \neq h(x))}\mathbb{E}_{x^-\sim p(\cdot|c^-)} e^{f(x)^T f(x^-)}} \right] \\
&= \mathbb{E}_{\mathcal{T}\sim\mathcal{D}} \mathbb{E}_{c\sim\rho_{\mathcal{T}};x\sim p(\cdot|c)} \left[-\log \frac{M e^{f(x)^T \mathbb{E}_{x^+\sim p(\cdot|c)} f(x^+)}}{M e^{f(x)^T \mathbb{E}_{x^+\sim p(\cdot|c)} f(x^+)} + M\cdot(K-1)\mathbb{E}_{\rho_{\mathcal{T}}(c^- | c^- \neq h(x))}\mathbb{E}_{x^-\sim p(\cdot|c^-)} e^{f(x)^T f(x^-)}} \right] \\
&\geq \mathbb{E}_{\mathcal{T}\sim\mathcal{D}} \mathbb{E}_{c\sim\rho_{\mathcal{T}};x\sim p(\cdot|c)} \left[-\log \frac{e^{f(x)^T \mathbb{E}_{x^+\sim p(\cdot|c)} f(x^+)}}{e^{f(x)^T \mathbb{E}_{x^+\sim p(\cdot|c)} f(x^+)} + (K-1)\mathbb{E}_{\rho_{\mathcal{T}}(c^- | c^- \neq h(x))} e^{f(x)^T \mathbb{E}_{x^-\sim p(\cdot|c^-)} f(x^-)}} \right] \\
&= \mathbb{E}_{\mathcal{T}\sim\mathcal{D}} \mathbb{E}_{c\sim\rho_{\mathcal{T}};x\sim p(\cdot|c)} \left[-\log \frac{\exp\big(f(x)^T \mu_c\big)}{\exp(f(x)^T\mu_c) + \sum_{c^-\in\mathcal{T}, c^-\neq c} \exp(f(x)^T \mu_{c^-})} \right] \\
&= \mathbb{E}_{\mathcal{T}\sim\mathcal{D}} L^{\mu}_{\text{Sup}}(\mathcal{T}, f) \\
&= \bar{L}^{\mu}_{\text{Sup}}(f)
\end{aligned}
$$

where the three inequalities follow from Jensen's inequality. The first and third inequality shift the expectations $\mathbb{E}_{x^+\sim p^+_{x,\mathcal{T}}}$ and $\mathbb{E}_{x^-\sim p(\cdot|c^-)}$, respectively, via the convexity of the functions and the second moves the expectation $\mathbb{E}_{\mathcal{T}\sim\mathcal{D}}$ out using concavity. Note that $\bar{L}_{\text{Sup}}(f) \leq \bar{L}^{\mu}_{\text{Sup}}(f)$ holds trivially. $\square$

B.3. Generalization bounds

We wish to derive a data dependent bound on the downstream supervised generalization error of the debiased contrastive objective. Recall that a sample $(x, \{x_i^+\}_{i=1}^M, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m)$ yields loss

$$
-\log\left\{ \frac{\sum_{i=1}^M e^{f(x)^\top f(x_i^+)}}{\sum_{i=1}^M e^{f(x)^\top f(x_i^+)} + M\cdot N \cdot G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m)} \right\} = \log\left\{ 1 + M\cdot N \frac{G(x, \{u_i\}_{i=1}^n, \{v_i\}_{i=1}^m)}{\sum_{i=1}^M e^{f(x)^\top f(x_i^+)}} \right\},
$$

which is equal to $\ell\left(\left\{ \frac{e^{f(x)^\top f(u_j)}}{\sum_{i=1}^M e^{f(x)^\top f(x_i^+)}} \right\}_{j=1}^n, \left\{ \frac{e^{f(x)^\top f(v_j)}}{\sum_{i=1}^M e^{f(x)^\top f(x_i^+)}} \right\}_{j=1}^m \right)$, where we define

$$
\begin{aligned}
\ell(\{a_i\}_{i=1}^n, \{b_i\}_{i=1}^m) &:= \log\left\{ 1 + M\cdot N \max\left(\frac{1}{\tau^-}\frac{1}{n}\sum_{i=1}^n a_i - \frac{\tau^+}{\tau^-}\frac{1}{m}\sum_{i=1}^m b_i, e^{-1} \right) \right\} \\
\widehat{L}^{M,N,n,m}_{\text{Debiased}}(f) &:= \frac{1}{T}\sum_{t=1}^T \ell\left(\left\{ \frac{e^{f(x_t)^\top f(u_{tj})}}{\sum_{i=1}^M e^{f(x_t)^\top f(x_{ti}^+)}} \right\}_{j=1}^n, \left\{ \frac{e^{f(x_t)^\top f(v_{tj})}}{\sum_{i=1}^M e^{f(x_t)^\top f(x_{ti}^+)}} \right\}_{j=1}^m \right) \\
\hat{f} &:= \arg\min_{f\in\mathcal{F}} \widehat{L}^{M,N,n,m}_{\text{Debiased}}(f)
\end{aligned}
$$

Theorem B.4. *With probability at least* $1-\delta$*, for all* $f \in \mathcal{F}$ *and* $N \geq K-1$*,*

$$L_{\text{Sup}}(\hat{f}) \leq L_{\text{Debiased}}^{M,N,n,m}(f) + \mathcal{O}\left(\frac{1}{\tau^-}\sqrt{\frac{1}{n}} + \frac{\tau^+}{\tau^-}\sqrt{\frac{1}{m}} + \frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log\frac{1}{\delta}}{T}}\right),$$

where $\lambda = \frac{1}{M}\sqrt{\frac{1}{\tau^{-2}}(\frac{m}{n}+1) + \tau^{+2}(\frac{n}{m}+1)}$ *and* $B = \log N\left(\frac{1}{\tau^-}+\tau^+\right)$*.*

Proof. Considering the samples to be $\left\{\left(x_t, \{x_{ti}^+\}_{i=1}^M, \{u_{ti}\}_{i=1}^n, \{v_{ti}\}_{i=1}^m\right)\right\}_{t=1}^T$. Then, we can use the standard bounds for empirical versus population means of any $B-$bounded function g belonging to a function class G, we have that with probability at least $1-\frac{\delta}{2}$.

$$\mathbb{E}[g(x)] \leq \frac{1}{T}\sum_{t=1}^T g(x_i) + \frac{2\mathcal{R}_S(G)}{T} + 3B\sqrt{\frac{\log\left(\frac{4}{\delta}\right)}{2T}} \tag{12}$$

In order to calculate $\mathcal{R}_S(G)$ we use the same trick as in (Saunshi et al., 2019). We express it as a composition of functions $g = \ell\left(h\left(f\left(x_t, \{x_{ti}^+\}_{i=1}^M, \{u_{ti}\}_{i=1}^n, \{v_{ti}\}_{i=1}^m\right)\right)\right)$ where $f \in \mathcal{F}$ just maps each sample to corresponding feature vector and h maps the feature vectors to the $\{a\}_{i=1}^n, \{b\}_{i=1}^m$. Then we use contraction inequality to bound $\mathcal{R}_S(G)$ with $\mathcal{R}_S(\mathcal{F})$. In order to do this we need to compute the Lipschitz constant for the intermediate function h in the composition.

For h, we see that the Jacobian has the following form

$$\frac{\partial a_i}{\partial f(x)} = a_i \frac{\sum_{j=1}^M (f(u_i) - f(x_j)) e^{f(x)^\top f(x_j^+)}}{\sum_{j=1}^M e^{f(x)^\top f(x_j^+)}}; \quad \frac{\partial b_i}{\partial f(x)} = b_i \frac{\sum_{j=1}^M (f(v_i) - f(x_j)) e^{f(x)^\top f(x_j^+)}}{\sum_{j=1}^M e^{f(x)^\top f(x_j^+)}}$$
$$\frac{\partial a_i}{\partial f(x_j^+)} = -a_i \frac{f(x) e^{f(x)^\top f(x_j^+)}}{\sum_{k=1}^M e^{f(x)^\top f(x_k^+)}}; \quad \frac{\partial b_i}{\partial f(x_j^+)} = -b_i \frac{f(x) e^{f(x)^\top f(x_j^+)}}{\sum_{k=1}^M e^{f(x)^\top f(x_k^+)}}$$
$$\frac{\partial a_i}{\partial f(u_j)} = f(x) a_i \delta(i-j); \quad \frac{\partial b_i}{\partial f(v_j)} = f(x) b_i \delta(i-j)$$

Using the fact that $\|f(\cdot)\|_2 = 1$, we get $\frac{e^{-2}}{M} \leq a_i, b_i \leq \frac{e^2}{M}$ and

$$\begin{aligned}
\|J\|_2^2 \leq \|J\|_F^2 \leq & \sum_{i=1}^n a_i^2 \left(\left\|\frac{\sum_{j=1}^M (f(u_i) - f(x_j)) e^{f(x)^\top f(x_j^+)}}{\sum_{j=1}^M e^{f(x)^\top f(x_j^+)}}\right\|_2^2 + \|f(x)\|_2^2 \frac{\sum_{j=1}^M e^{2f(x)^\top f(x_j^+)}}{\left(\sum_{j=1}^M e^{f(x)^\top f(x_j^+)}\right)^2} + \|f(x)\|_2^2\right) \\
& + \sum_{i=1}^m b_i^2 \left(\left\|\frac{\sum_{j=1}^M (f(v_i) - f(x_j)) e^{f(x)^\top f(x_j^+)}}{\sum_{j=1}^M e^{f(x)^\top f(x_j^+)}}\right\|_2^2 + \|f(x)\|_2^2 \frac{\sum_{j=1}^M e^{2f(x)^\top f(x_j^+)}}{\left(\sum_{j=1}^M e^{f(x)^\top f(x_j^+)}\right)^2} + \|f(x)\|_2^2\right) \\
& \leq \sum_{i=1}^n a_i^2 (4+1+1) + \sum_{i=1}^m b_i^2 (4+1+1) \leq \frac{6(n+m)e^4}{M^2}
\end{aligned}$$

Using this and the Lipschitz constant, $O\left(\sqrt{\frac{1}{n\tau^{-2}} + \frac{\tau^{+2}}{m}}\right)$ of ℓ derived in (Chuang et al., 2020), we get $\mathcal{R}_S(\mathcal{G}) = \lambda \mathcal{R}_S(\mathcal{F})$ where $\lambda = \mathcal{O}\left(\frac{1}{M}\sqrt{\frac{1}{\tau^{-2}}(\frac{m}{n}+1) + \tau^{+2}(\frac{n}{m}+1)}\right)$. From (Chuang et al., 2020), we also get $B = O\left(\log N\left(\frac{1}{\tau^-}+\tau^+\right)\right)$. Combining this with Equation 12 gives us that with probability at least $1-\frac{\delta}{2}$

$$L_{\text{Debiased}}^{M,N,n,m}(\hat{f}) \leq \widehat{L}_{\text{Debiased}}^{M,N,n,m}(\hat{f}) + \mathcal{O}\left(\frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log\frac{1}{\delta}}{T}}\right)$$

Using Theorem B.2, we get that

$$
\begin{aligned}
L^{M,N}_{\text{Unbiased}}(\hat{f}) &\leq L^{M,N,n,m}_{\text{Debiased}}(\hat{f}) + \mathcal{O}\left(\frac{1}{\tau^-}\sqrt{\frac{1}{n}} + \frac{\tau^+}{\tau^-}\sqrt{\frac{1}{m}}\right) \\
&\leq \widehat{L}^{M,N,n,m}_{\text{Debiased}}(\hat{f}) + \mathcal{O}\left(\frac{1}{\tau^-}\sqrt{\frac{1}{n}} + \frac{\tau^+}{\tau^-}\sqrt{\frac{1}{m}} + \frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log\frac{1}{\delta}}{T}}\right)
\end{aligned}
$$

Using Lemma B.3, we get

$$
L_{\text{Sup}}(\hat{f}) \leq L^{M,N}_{\text{Unbiased}}(\hat{f}) \leq \widehat{L}^{M,N,n,m}_{\text{Debiased}}(\hat{f}) + \mathcal{O}\left(\frac{1}{\tau^-}\sqrt{\frac{1}{n}} + \frac{\tau^+}{\tau^-}\sqrt{\frac{1}{m}} + \frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log\frac{1}{\delta}}{T}}\right)
$$

Finally we see that for any f, we can use M Hoeffding's inequality to show that with at least $1-\frac{\delta}{2}$ probability

$$
\widehat{L}^{M,N,n,m}_{\text{Debiased}}(f) \leq L^{M,N,n,m}_{\text{Debiased}}(f) + 3B\sqrt{\frac{\log\left(\frac{2}{\delta}\right)}{2T}}
$$

Combining all of the above results gives us that with probability at least $1-\delta$,

$$
\begin{aligned}
L_{\text{Sup}}(\hat{f}) \leq L^{M,N}_{\text{Unbiased}}(\hat{f}) &\leq \widehat{L}^{M,N,n,m}_{\text{Debiased}}(\hat{f}) + \mathcal{O}\left(\frac{1}{\tau^-}\sqrt{\frac{1}{n}} + \frac{\tau^+}{\tau^-}\sqrt{\frac{1}{m}} + \frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log\frac{1}{\delta}}{T}}\right) \\
&\leq \widehat{L}^{M,N,n,m}_{\text{Debiased}}(f) + \mathcal{O}\left(\frac{1}{\tau^-}\sqrt{\frac{1}{n}} + \frac{\tau^+}{\tau^-}\sqrt{\frac{1}{m}} + \frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log\frac{1}{\delta}}{T}}\right) \\
&\leq L^{M,N,n,m}_{\text{Debiased}}(f) + \mathcal{O}\left(\frac{1}{\tau^-}\sqrt{\frac{1}{n}} + \frac{\tau^+}{\tau^-}\sqrt{\frac{1}{m}} + \frac{\lambda \mathcal{R}_{\mathcal{S}}(\mathcal{F})}{T} + B\sqrt{\frac{\log\frac{1}{\delta}}{T}}\right) + \mathcal{O}\left(B\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{T}}\right)
\end{aligned}
$$

□

C. Table of Definitions

Table S1. A summary of definitions.

$\mathcal{L}_{\text{NCA}}(G^1, M)$	$\mathbb{E}_{x\sim\mathcal{D},x_j^+\sim\mathcal{D}_x^{\text{aug}},x_i^-\sim\mathcal{D}_{\setminus x}^{\text{aug}}}[-\log \frac{\sum_{j=1}^{M} e^{f(x)^T f(x_j^+)}}{\sum_{j=1}^{M} e^{f(x)^T f(x_j^+)}+NG^1(x,\{x_i^-\}^N)}]$
$\mathcal{L}_{\text{MIXNCA}}(G^1, M, \lambda)$	$\mathbb{E}_{x\sim\mathcal{D},x^+\sim\mathcal{D}_x^{\text{aug}},x_{i_1}^-,x_{i_2j}^-,x_j^-\sim\mathcal{D}_{\setminus x}^{\text{aug}}}[-\log \frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)}+NG^1(x,\{x_{i_1}^-\}^N)}$ $-\frac{\lambda}{M-1}\sum_{j=1}^{M-1}\log \frac{e^{f(x)^T f(\lambda x^+ +(1-\lambda)x_j^-)}}{e^{f(x)^T f(\lambda x^+ +(1-\lambda)x_j^-)}+NG^1(x,\{x_{i_2j}^-\}_{i_2}^N)}$ $-\frac{1-\lambda}{M-1}\sum_{j=1}^{M-1}\log\left(1-\frac{e^{f(x)^T f(\lambda x^+ +(1-\lambda)x_j^-)}}{e^{f(x)^T f(\lambda x^+ +(1-\lambda)x_j^-)}+NG^1(x,\{x_{i_2j}^-\}_{i_2}^N)}\right)]$
$g_0(x, \{x_i^-\}_i^N)$	$\frac{1}{N}\sum_{i=1}^{N} e^{f(x)^T f(x_i^-)}$
$g_1(x, \{u_i\}^n, \{v_j\}^m)$	$\max\{\frac{1}{1-\tau^+}(\frac{1}{n}\sum_{i=1}^{n} e^{f(x)^T f(u_i)} - \tau^+ \frac{1}{m}\sum_{j=1}^{m} e^{f(x)^T f(v_j)}), e^{-1/t}\}$
$g_2(x, \{u_i\}^n, \{v_j\}^m)$	$\max\{\frac{1}{1-\tau^+}(\frac{\sum_{i=1}^{n} e^{(\beta+1)f(x)^T f(u_i)}}{\sum_{i=1}^{n} e^{\beta f(x)^T f(u_i)}} - \tau^+ \frac{1}{m}\sum_{j=1}^{m} e^{f(x)^T f(v_j)}), e^{-1/t}\}$
$\hat{w}(x)$	$-\log \frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)}+NG(x,\cdot)}$

D. Complete Tables of Results

We give the full table of results in Section 4 in the following. Notably, we gather the standard accuracy, robust accuracy, transfer accuracy, and transfer robust accuracy for each specification.

Table S2. 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.

M	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_0, M)$			
	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	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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_0, M, 0.5)$			
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_0, M, 0.6)$			
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_0, M, 0.7)$			
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**
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_0, M, 0.8)$			
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**
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_0, M, 0.9)$			
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 S3. 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.

M	CIFAR100 Acc.	FGSM Acc.	CIFAR10 Acc.	FGSM Acc.
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, M)$			
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$			
1	56.83±0.20	31.03±0.41	77.24±0.29	48.38±0.70
2	59.41±0.19	**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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.6)$			
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.7)$			
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.8)$			
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.9)$			
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 S4. 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.

M	CIFAR100 Acc.	FGSM Acc.	CIFAR10 Acc.	FGSM Acc.
	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, M)$			
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
	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$			
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**
	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.6)$			
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
	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.7)$			
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
	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.8)$			
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
	$\alpha \neq 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.9)$			
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

E. Robust Accuracy

For a more comprehensive study of adversarial robustness, we extend Table S3 to include PGD attack results with the same strength as FGSM attacks ($\epsilon = 0.002$). One can readily see from Table S5 that the robust 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 S5. The complete Table S3 (Table 1 right column) with additional PGD accuracy.

M	CIFAR100 Acc.	FGSM Acc.	PGD Acc.	CIFAR10 Acc.	FGSM Acc.	PGD Acc.
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{NCA}}(g_2, M)$					
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.5)$					
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	59.41±0.19	**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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.6)$					
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.7)$					
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.8)$					
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
	$\alpha = 0,\ \mathcal{L}_{\text{NaCl}}(G^1, M, \lambda) = \mathcal{L}_{\text{MIXNCA}}(g_2, M, 0.9)$					
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 robust 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 robust accuracy of our proposals (IntCl & IntNaCl) together with baselines under all attacks. From Figure S1, one can see that among all baselines, Adv demonstrates the best adversarial robustness, whereas our proposals still consistently win over it by a noticeable margin.

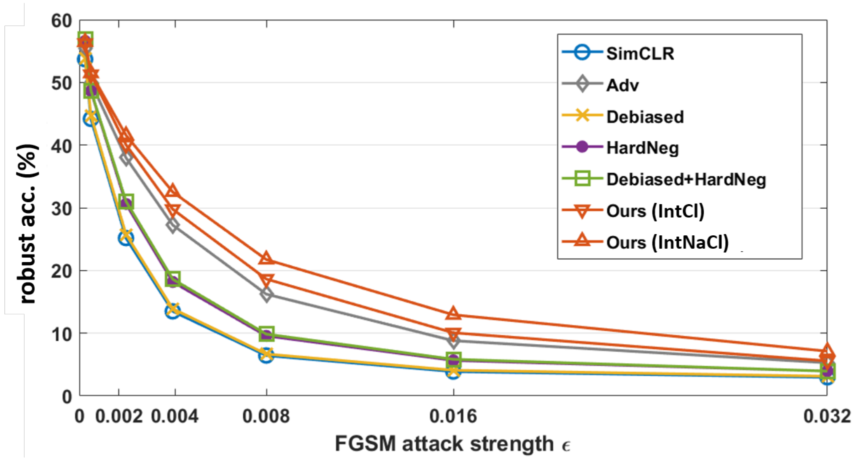
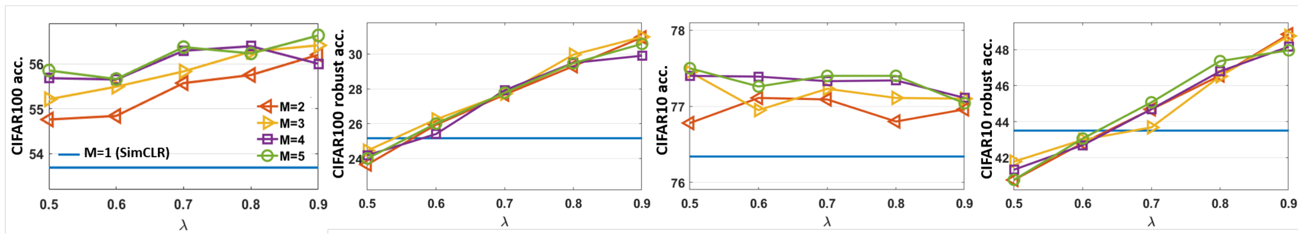
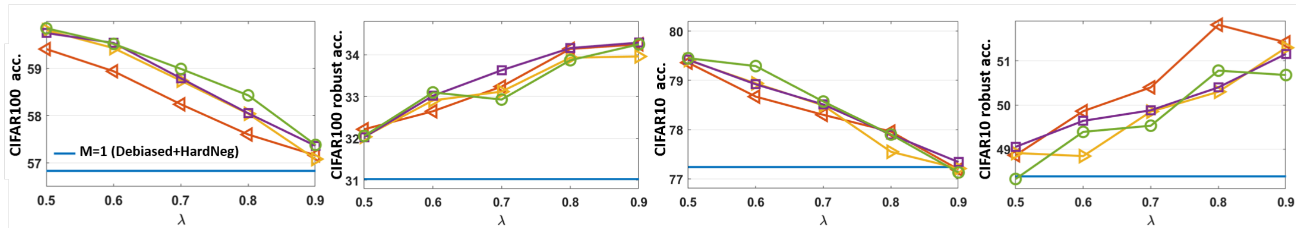


Figure S1. The robust accuracy under FGSM attacks of different strength on CIFAR100.

F. The Effect of λ



(a) NaCl on SimCLR (Chen et al., 2020a), i.e. $\alpha = 0, \mathcal{L}_{\text{NaCl}} = \mathcal{L}_{\text{MIXNCA}}, G^1 = g_0$ in Eq. (9)



(b) NaCl on Debiased+HardNeg (Robinson et al., 2021), i.e. $\alpha = 0, \mathcal{L}_{\text{NaCl}} = \mathcal{L}_{\text{MIXNCA}}, G^1 = g_2$ in Eq. (9)

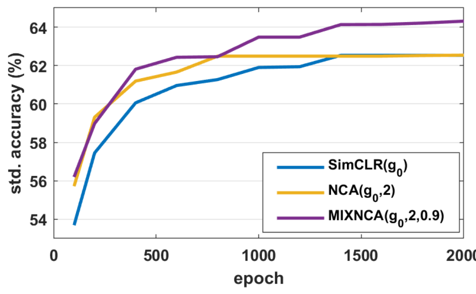
Figure S2. The standard and robust accuracy (%) on CIFAR100 and CIFAR10 as functions of λ in Eq. (9) when $\alpha = 0, \mathcal{L}_{\text{NaCl}} = \mathcal{L}_{\text{MIXNCA}}$.

G. Extended Runtime

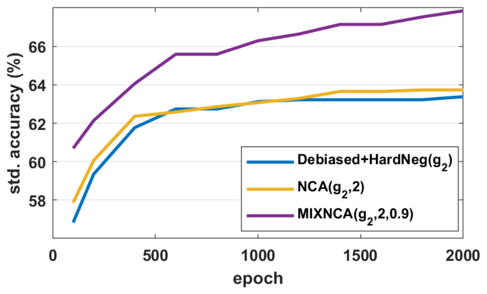
As training the representation with more epochs can also expose the data to more augmentations, we carry out an additional experiments to compare the efficiency and ultimate accuracy of $\mathcal{L}_{\text{NaCl}}$, $\mathcal{L}_{\text{SimCLR}}$, and $\mathcal{L}_{\text{Debiased+HardNeg}}$. In Table S6, we give the standard accuracy of NaCl on SimCLR and NaCl on Debiased+HardNeg at different epochs. Same as before, we only select one λ when $\mathcal{L}_{\text{NaCl}} = \mathcal{L}_{\text{MIXNCA}}$ and report its results together with those of $\mathcal{L}_{\text{NaCl}} = \mathcal{L}_{\text{NCA}}$. In Figure S3, we plot the best standard accuracy achieved as a function of training epochs. Specially, (HaoChen et al., 2021) has reported a $\mathcal{L}_{\text{SimCLR}}$ CIFAR100 accuracy of 54.74% after 200 epochs, compared to $\mathcal{L}_{\text{NCA}}(g_0, 2)$'s 55.72% after 100 epochs. In our reproduction of the $\mathcal{L}_{\text{SimCLR}}$ 200-epoch result[2], 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 $\mathcal{L}_{\text{NCA}}(g_0, 2)$ takes 158 mins). Overall, we see that NaCl methods demonstrate better efficiency when applying on SimCLR and better ultimate accuracy when applying on Debiased+HardNeg.

#epoch	100	200	400	600	800	1000	1200	1400	1600	1800	2000
$\mathcal{L}_{\text{SimCLR}}$	53.69	57.45	60.06	60.96	61.27	61.90	61.94	62.53	62.44	62.10	62.06
$\mathcal{L}_{\text{NCA}}(g_0, 2)$	55.72	59.31	61.19	61.66	62.49	61.95	62.06	62.39	62.39	62.52	62.54
$\mathcal{L}_{\text{MIXNCA}}(g_0, 2, 0.9)$	56.20	58.98	61.81	62.43	62.46	63.48	63.48	64.13	64.14	64.21	64.31
$\mathcal{L}_{\text{Debiased+HardNeg}}$	56.83	59.35	61.77	62.74	62.68	63.12	63.22	63.08	62.86	62.90	63.38
$\mathcal{L}_{\text{NCA}}(g_2, 2)$	57.87	60.06	62.36	62.58	62.86	63.07	63.29	63.65	63.13	63.73	63.20
$\mathcal{L}_{\text{MIXNCA}}(g_2, 2, 0.5)$	59.41	62.14	64.06	65.59	65.53	66.29	66.64	67.14	66.94	67.53	67.85

Table S6. The CIFAR100 linear evaluation results (%) after different numbers of training epochs.



(a) NaCl on SimCLR (Chen et al., 2020a)



(b) NaCl on Debiased+HardNeg (Robinson et al., 2021)

Figure S3. The standard accuracy (%) on CIFAR100 with extended runtime.

[2]We let the dataloader shuffle the whole dataset to form new batches after every epoch, so by doubling the training epoch, one will effectively expose the network to more diverse negative pairs.

H. Experimental Details

Architecture. We follow (Chen et al., 2020a; Robinson et al., 2021) to incorporate an MLP projection head during the contrastive learning on resnet18.

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

Training epochs. The representation network is trained for 100 epochs. For CIFAR100 and CIFAR10, the downstream fully-connected layer is trained for 1000 epochs. For TinyImagenet, the fully-connected layer is trained for 200 epochs.

Methodological hyperparameters. Throughout out experiments, we use $\tau^+ = 0.01$ and $\beta = 1.0$ for $\mathcal{L}_{\text{Debiased}}$ (Chuang et al., 2020) and $\mathcal{L}_{\text{Debiased+HardNeg}}$ (Robinson et al., 2021), $\alpha = 1$ for $\mathcal{L}_{\text{Adv}}$ (Ho & Vasconcelos, 2020). 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 (Wong et al., 2020), 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.

I. Supervised Learning Baseline

We give in the following the standard and robust 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 robust accuracy as described in Section 4.

CIFAR100 (std. acc., FGSM acc., PGD acc.): 65.16$\pm$0.32, 35.89$\pm$0.23, 32.62$\pm$0.23.

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