Title
Tuned Contrastive Learning

Author
Animesh, Chaitanya

Publication Date
2023

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Tuned Contrastive Learning

A thesis submitted in partial satisfaction of the
requirements for the degree of Master of Science

in

Computer Science

by

Chaitanya Animesh

Committee in charge:

Professor Manmohan Krishna Chandraker, Chair
Professor Taylor Darwin Berg-Kirkpatrick
Professor Garrison W. Cottrell

2023

The thesis of Chaitanya Animesh is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

To my mother, father, sister and Lord Rama — the ultimate consciousness.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I am grateful to Prof. Manmohan Chandraker, for his support and advising during my research work. Without his unwavering support, this thesis would not have been possible. I also thank Prof. Gary Cottrell and Prof. Taylor Berg-Kirkpatrick for teaching ‘CSE 251B’ (course on deep learning) and ‘CSE 251A’ (course on machine learning) respectively in the Winter’22 quarter. These courses helped me begin my journey in deep learning research. I also thank them for being part of my thesis committee.

I thank Tarun Kalluri who helped me during my thesis research work.

I thank all my family members, without whose constant support, this work would not have been possible.

This thesis, in its entirety, has been submitted for publication with the possibility of its material being included in a conference in 2023, authored by Chaitanya Animesh and Manmohan Chandraker. The thesis author was the primary investigator and the first author of this paper.

ABSTRACT OF THE THESIS

Tuned Contrastive Learning

by

Chaitanya Animesh

Master of Science in Computer Science

University of California San Diego, 2023

Professor Manmohan Krishna Chandraker, Chair

In recent times, contrastive learning has become increasingly popular for visual self-supervised representation learning owing to their state-of-the-art (SOTA) performance. Most of the modern contrastive learning methods generalize only to one positive and multiple negatives per anchor. A recent state-of-the-art, supervised contrastive (SupCon) loss, extends self-supervised contrastive learning to supervised setting by generalizing to multiple positives and negatives in a batch and improves upon the cross-entropy loss. In this thesis, we propose a novel contrastive loss function – Tuned Contrastive Learning (TCL) loss, that generalizes to multiple positives and negatives in a batch and offers parameters to tune and improve the gradient responses from hard positives and hard negatives. We provide theoretical analysis of our

loss function's gradient response and show mathematically how it is better than that of SupCon loss. We empirically compare our loss function with SupCon loss and cross-entropy loss in a supervised setting on multiple classification-task datasets to show its effectiveness. We also show the stability of our loss function to a range of hyperparameter settings. Unlike SupCon loss that is only applied to supervised setting, we show how to extend TCL to self-supervised setting and empirically compare it with various SOTA self-supervised learning methods. Hence, we show that TCL achieves performance on par with SOTA methods in both supervised and self-supervised settings.

Chapter 1

Introduction

Paucity of labeled data limits the application of supervised learning to various visual learning tasks [36]. As a result, unsupervised [17, 19, 26] and self-supervised based learning methods [6, 33, 18, 4] have garnered a lot of attention and popularity for their ability to learn from vast unlabeled data. Such methods can be broadly classified into two categories: generative methods and discriminative methods. Generative methods [17, 26] train deep neural networks to generate in the input space i.e. the pixel space and hence, are computationally expensive and not necessary for representation learning. On the other hand, discriminative approaches [16, 13, 34, 30, 1, 6] train deep neural networks to learn representations for pretext tasks using unlabeled data and an objective function. Out of these discriminative based approaches, contrastive learning based methods [30, 1, 6] have performed significantly well and are an active area of research.

The common principle of contrastive learning based methods in an unsupervised setting is to create semantic preserving transformations of each sample which are called *positives* and treat transformations of other samples in a training batch as *negatives* [2, 23]. The contrastive loss objective considers every transformed sample as a reference sample, called an *anchor*, and is then used to train the network architecture to pull the positives (for that anchor) closer to the anchor and push the negatives away from the anchor in latent space [2, 23]. The positives are often created using various data augmentation strategies. Supervised Contrastive Learning [23]

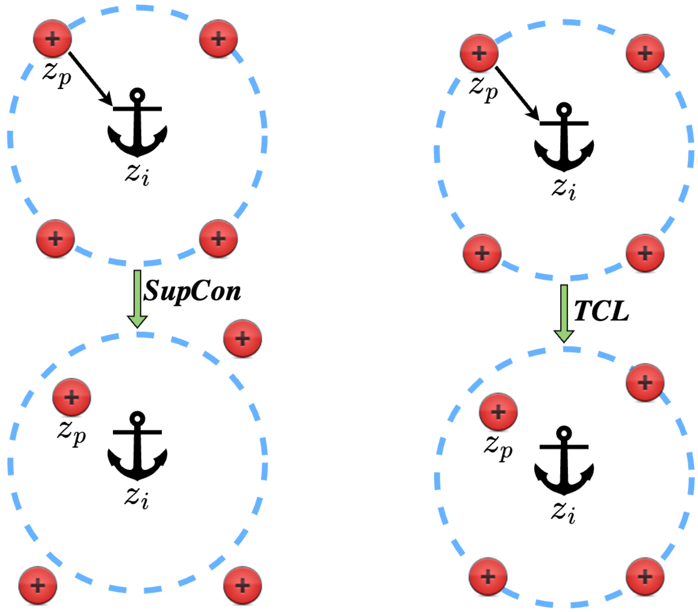

Figure 1.1. Figure illustrates intuitively how TCL loss differs from SupCon loss [23]. For the SupCon loss per sample — L_i^{sup} (equation 3.2) to decrease, the anchor z_i will pull the positive z_p but push away the other positives to some extent in the embedding space. TCL loss reduces this effect.

extended contrastive learning to supervised setting by using the label information and treating the other samples in the batch having the same label as that of the anchor also as positives in addition to the ones produced through data augmentation strategies. It presents a new loss called supervised contrastive loss (abbreviated as SupCon loss) that can be viewed as a loss generalizing to multiple positives available in a batch.

In this work, we propose a novel contrastive learning loss objective, which we call **Tuned Contrastive Learning (TCL) Loss** that can use multiple positives and multiple negatives present in a batch. We show how it can be used in supervised as well as self-supervised settings. TCL loss improves upon the limitations of the SupCon loss: 1. Implicit consideration of positives as negatives and, 2. No provision for regulating hard negative gradient response. TCL loss thus gives better gradient response to hard positives and hard negatives. This leads to small ($< 1\%$ in terms of classification accuracy) but consistent improvements in performance over SupCon loss and outperformance over cross-entropy loss. Since TCL generalizes to multiple positives, we then present a novel idea of having and using positive triplets (and possibly more) instead of being limited to positive pairs for self-supervised learning. We evaluate our loss function in self-supervised settings without making use of any label information and show how TCL outperforms SimCLR [6] and performs on par with various SOTA self-supervised learning methods [18, 33, 36, 4, 20, 8, 15, 9, 3, 37]. The key highlights of the thesis are as follows:

1. We identify and analyse in detail two limitations of the supervised contrastive (SupCon) loss.

2. We present a novel contrastive loss function called Tuned Contrastive Learning (TCL) loss that generalizes to multiple positives and multiple negatives in a batch, overcomes the described limitations of the SupCon loss and is applicable in both supervised and self-supervised settings. We mathematically show with clear proofs how our loss's gradient response is better than that of SupCon loss.

3. We compare TCL loss with SupCon loss (as well as cross-entropy loss) in a supervised setting on various classification-task datasets and show that TCL loss gives consistent improvements in top-1 accuracy over SupCon loss. We empirically show the stability of TCL loss over a range of hyperparameters: network architecture, batch size, projector size and augmentation strategy.

4. At last, we present a novel idea of having positive triplets (and possibly more) instead of positive pairs and show how TCL can be extended to self-supervised settings. We empirically show that TCL outperforms SimCLR, and performs on par with various SOTA self-supervised learning (SSL) methods.

Chapter 2

Related Work

In this chapter, we cover various popular and recent works in brief involving contrastive learning.

Deep metric learning methods originated with the idea of contrastive losses and were introduced with the goal of learning a distance metric between samples in a high-dimensional space [2]. The goal in such methods is to learn a function that maps similar samples to nearby points in this space, and dissimilar samples to distant points. There is often a margin parameter, m, imposing the distance between examples from different classes to be larger than this value [2]. The triplet loss [22] and the proposed improvements [7, 27] on it used this principle. These methods rely heavily on sophisticated sampling techniques for choosing samples in every batch for better training.

SimCLR [6], an Info-NCE loss [30] based framework, learns visual representations by increasing the similarity between the embeddings of two augmented views of the input image. Augmented views generally come from a series of transformations like random resizing, cropping, color jittering, and random blurring. Although they make use of multiple negatives, only one positive is available per anchor. They require large batch sizes in order to have more hard negatives in the batch to learn from and boost the performance. SupCon loss [23] applies contrastive learning in supervised setting by basically extending the SimCLR loss to generalize to multiple positives available in a batch and improves upon the cross-entropy loss which lacks robustness to noisy labels [35, 29] and has the possibility of poor margins [25, 14].

Unlike SimCLR or SupCon, many SOTA SSL approaches only work with positives (don't require negatives) or use different approaches altogether. BYOL [18] uses asymmetric networks with one network using an additional predictor module while the other using exponential moving average (EMA) to update its weights, in order to learn using positive pairs only and prevent collapse. SimSiam [9] uses stop-gradient operation instead of EMA and asymmetric networks to achieve the same goal. Barlow Twins [33] objective function on the other hand computes the cross-correlation matrix between the embeddings of two identical networks fed with augmentations of a batch of samples, and tries to make this matrix close to identity. SwAV uses a clustering approach and enforces consistency between the cluster assignments of multiple positives produced through multi-crop strategy [4].

Chapter 3

Supervised Contrastive Learning and its Issues

The framework for Supervised Contrastive Learning consists of three components: a data augmentation module that produces two augmentations for each sample in the batch, an encoder network that maps the augmentations to their corresponding representation vectors and a projection network that produces normalized embeddings for the representation vectors to be fed to the loss function. The projection network is later discarded and the encoder network is used at inference time by training a linear classifier (attached to the frozen encoder) with cross-entropy loss. Section 3.1 of [23] contains more details on this. The SupCon loss is given by the following two equations (refers to L_{out}^{sup} in [23]):

$$L^{sup} = \sum_{i \in I} L_i^{sup} \tag{3.1}$$

where

$$L_i^{sup} = \frac{-1}{|P(i)|} \sum_{p \in P(i)} log(\frac{exp(z_i.z_p/\tau)}{\sum_{p' \in P(i)} exp(z_i.z_{p'}/\tau) + \sum_{n \in N(i)} exp(z_i.z_n/\tau)}) \tag{3.2}$$

Here I denotes the batch of samples obtained after augmentation and so, will be twice the size of the original input batch. $i \in I$ denotes a sample (anchor) within it. z_i denotes the normalized projection network embedding of the sample i as given by the projector network. $P(i)$ is the set of all positives for the anchor i (except the anchor i itself) i.e. positive from the augmentation module and positives with the same label as anchor i in the batch I. $N(i)$ denotes the set of negatives in the batch such that $N(i) \equiv I \setminus (P(i) \cup \{i\})$. As shown in Section 2 of the supplementary material of [23], we have the following lemma:

Lemma 1. *The gradient of the SupCon loss per sample — L_i^{sup} with respect to the normalized projection network embedding z_i is given by:*

$$\frac{\partial L_i^{sup}}{\partial z_i} = \frac{1}{\tau}(\underbrace{\sum_{p \in P(i)} z_p (P_{ip}^s - X_{ip})}_{\textit{Gradient response from positives}} + \underbrace{\sum_{n \in N(i)} z_n P_{in}^s}_{\textit{Gradient response from negatives}}) \tag{3.3}$$

where

$$X_{ip} = \frac{1}{|P(i)|} \tag{3.4}$$

$$P_{ip}^s = \frac{\exp(z_i . z_p / \tau)}{\sum_{a \in A(i)} \exp(z_i . z_a / \tau)} \tag{3.5}$$

$$P_{in}^s = \frac{\exp(z_i . z_n / \tau)}{\sum_{a \in A(i)} \exp(z_i . z_a / \tau)} \tag{3.6}$$

Note that $A(i) \equiv P(i) \cup N(i)$ here. The authors further show in Section 3 of the supplementary [23] that the gradient from a positive while flowing back through the projector into the encoder reduces to almost zero for easy positives and $|P_{ip}^s - X_{ip}|$ for a hard positive because of the normalization consideration in the projection network. Similarly, the gradient from a negative reduces to almost zero for easy negatives and P_{in}^s for a hard negative. We now present and analyse the following two limitations of the SupCon loss:

1. **Implicit consideration of positives as negatives**: Having a closer look at the L_i^{sup} (equation 3.2) loss term reveals that the numerator inside the log function considers similarity with one positive p at a time while the denominator considers similarity of the anchor i with all the positives in the batch — the set $P(i)$, thereby implicitly considering all the positives as negatives. A glance at the derivation of Lemma 1 in [23] clearly shows that

this leads to the magnitude of the gradient response from a hard positive getting reduced to $|X_{ip} - P^s_{ip}|$ instead of simply $|X_{ip}|$. The term P^s_{ip} consists of an exponential term in the numerator and thus can reduce the magnitude of $|X_{ip} - P^s_{ip}|$ considerably, especially because the temperature τ is generally chosen to be small. Note that the authors of [23] approximate the numerator of P^s_{ip} to 1 while considering the magnitude of $|X_{ip} - P^s_{ip}|$ in their supplementary by assuming $z_i.z_p \approx 0$ for a hard positive which might not always be true. Another way to look at this limitation analytically is to observe the log part in the L^{sup}_i loss term. For the loss term to decrease and ideally converge to close to zero, the numerator term inside the log function will encourage the anchor z_i to pull the positive z_p towards it while the denominator term will encourage it to push away the other positives present in $P(i)$ by some extent, thereby treating the other positives as negatives implicitly.

2. **No provision for regulating P^s_{in}:** The authors of [6, 23] mention that performance in contrastive learning benefits from hard negatives and that gradient contribution from hard negatives should be higher. It is easy to observe from equation 3.6 that the magnitude of the gradient signal from a hard negative — P^s_{in} in the SupCon loss decreases with batch size and the number of positives in the batch, and can become considerably small, especially since the denominator consists of terms denoting the similarity between the anchor and all the positives in the batch which are temperature scaled and exponentiated. This can limit the gradient contribution from hard negatives.

Chapter 4

Tuned Contrastive Learning

In this chapter, we present our novel contrastive loss function — **Tuned Contrastive Learning (TCL) Loss**. Note that our representation learning framework remains the same as that of Supervised Contrastive Learning discussed before. The TCL loss is given by the following equations:

$$L^{tcl} = \sum_{i \in I} L_i^{tcl} \tag{4.1}$$

$$L_i^{tcl} = \frac{-1}{|P(i)|} \sum_{p \in P(i)} log(\frac{exp(z_i . z_p / \tau)}{D(z_i)}) \tag{4.2}$$

where

$$D(z_i) = \sum_{p' \in P(i)} exp(z_i . z_{p'} / \tau) + k_1 (\sum_{p' \in P(i)} exp(-z_i . z_{p'})) + k_2 (\sum_{n \in N(i)} exp(z_i . z_n / \tau)) \tag{4.3}$$

$$k_1, k_2 \geq 1 \tag{4.4}$$

k_1 and k_2 are scalar parameters that are fixed before training. All other symbols have the same meaning as discussed in the previous section. We now present the following lemma.

Lemma 2. *The gradient of the TCL loss per sample —* L_i^{tcl} *with respect to the normalized projection network embedding* z_i *is given by:*

$$\frac{\partial L_i^{tcl}}{\partial z_i} = \frac{1}{\tau}\Big(\underbrace{\sum_{p\in P(i)} z_p(P_{ip}^t - X_{ip} - Y_{ip}^t)}_{\textit{Gradient response from positives}} + \underbrace{\sum_{n\in N(i)} z_n P_{in}^t}_{\textit{Gradient response from negatives}}\Big) \tag{4.5}$$

where

$$X_{ip} = \frac{1}{|P(i)|} \tag{4.6}$$

$$P_{ip}^t = \frac{exp(z_i.z_p/\tau)}{D(z_i)} \tag{4.7}$$

$$Y_{ip}^t = \frac{\tau k_1 exp(-z_i.z_p)}{D(z_i)} \tag{4.8}$$

$$P_{in}^t = \frac{k_2 exp(z_i.z_n/\tau)}{D(z_i)} \tag{4.9}$$

Proof.

$$L_i^{tcl} = \frac{-1}{|P(i)|}\sum_{p\in P(i)} log\Big(\frac{exp(z_i.z_p/\tau)}{D(z_i)}\Big) \tag{4.10}$$

$$\implies L_i^{tcl} = \frac{-1}{|P(i)|}\sum_{p\in P(i)} \big(\frac{z_i.z_p}{\tau} - log(D(z_i))\big) \tag{4.11}$$

$$\begin{aligned}\implies \frac{\partial L_i^{tcl}}{\partial z_i} = \frac{-1}{\tau|P(i)|}\sum_{p\in P(i)}\Bigg(z_p - \frac{(\sum_{p'\in P(i)} z_{p'} exp(z_i.z_{p'}/\tau)}{D(z_i)} \\ + \frac{\tau k_1(\sum_{p'\in P(i)} z_{p'} exp(-z_i.z_{p'}))}{D(z_i)} - \frac{k_2(\sum_{n\in N(i)} z_n exp(z_i.z_n/\tau))}{D(z_i)}\Bigg)\end{aligned} \tag{4.12}$$

$$\begin{aligned}\implies \frac{\partial L_i^{tcl}}{\partial z_i} = \frac{-1}{\tau|P(i)|}\Bigg[\sum_{p\in P(i)} z_p - \sum_{p\in P(i)}\frac{(\sum_{p'\in P(i)} z_{p'} exp(z_i.z_{p'}/\tau))}{D(z_i)} \\ + \sum_{p\in P(i)}\frac{\tau k_1(\sum_{p'\in P(i)} z_{p'} exp(-z_i.z_{p'}))}{D(z_i)} - \sum_{p\in P(i)}\frac{k_2(\sum_{n\in N(i)} z_n exp(z_i.z_n/\tau))}{D(z_i)}\Bigg]\end{aligned} \tag{4.13}$$

$$\implies \frac{\partial L_i^{tcl}}{\partial z_i} = \frac{-1}{\tau|P(i)|}\Bigg[\sum_{p\in P(i)} z_p - \sum_{p'\in P(i)} \frac{(\sum_{p\in P(i)} z_{p'} exp(z_i.z_{p'}/\tau))}{D(z_i)}$$
$$+ \sum_{p'\in P(i)} \frac{\tau k_1(\sum_{p\in P(i)} z_{p'} exp(-z_i.z_{p'}))}{D(z_i)} - \sum_{p\in P(i)} \frac{k_2(\sum_{n\in N(i)} z_n exp(z_i.z_n/\tau))}{D(z_i)}\Bigg] \tag{4.14}$$

$$\implies \frac{\partial L_i^{tcl}}{\partial z_i} = \frac{-1}{\tau|P(i)|}\Bigg[\sum_{p\in P(i)} z_p - \sum_{p'\in P(i)} \frac{(|P(i)| z_{p'} exp(z_i.z_{p'}/\tau))}{D(z_i)}$$
$$+ \sum_{p'\in P(i)} \frac{\tau k_1(|P(i)| z_{p'} exp(-z_i.z_{p'}))}{D(z_i)} - \frac{|P(i)| k_2(\sum_{n\in N(i)} z_n exp(z_i.z_n/\tau))}{D(z_i)}\Bigg] \tag{4.15}$$

$$\implies \frac{\partial L_i^{tcl}}{\partial z_i} = \frac{-1}{\tau|P(i)|}\Bigg[\sum_{p\in P(i)} z_p - \sum_{p\in P(i)} \frac{(|P(i)| z_p exp(z_i.z_p/\tau))}{D(z_i)}$$
$$+ \sum_{p\in P(i)} \frac{\tau k_1(|P(i)| z_p exp(-z_i.z_p))}{D(z_i)} - \frac{|P(i)| k_2(\sum_{n\in N(i)} z_n exp(z_i.z_n/\tau))}{D(z_i)}\Bigg] \tag{4.16}$$

$$\implies \frac{\partial L_i^{tcl}}{\partial z_i} = \frac{-1}{\tau}\Bigg[\sum_{p\in P(i)} \frac{z_p}{|P(i)|} - \sum_{p\in P(i)} \frac{(z_p exp(z_i.z_p/\tau))}{D(z_i)}$$
$$+ \sum_{p\in P(i)} \frac{\tau k_1(z_p exp(-z_i.z_p))}{D(z_i)} - \frac{k_2(\sum_{n\in N(i)} z_n exp(z_i.z_n/\tau))}{D(z_i)}\Bigg] \tag{4.17}$$

$$\implies \frac{\partial L_i^{tcl}}{\partial z_i} = \frac{1}{\tau}\Bigg[\sum_{p\in P(i)} z_p\left(\frac{exp(z_i.z_p/\tau)}{D(z_i)} - \frac{1}{|P(i)|} - \frac{\tau k_1 exp(-z_i.z_p)}{D(z_i)}\right)$$
$$+ \sum_{n\in N(i)} z_n \frac{k_2 exp(z_i.z_n/\tau)}{D(z_i)}\Bigg] \tag{4.18}$$

This completes the proof. □

From Lemma 2, Theorem 1 and Theorem 2 follow in a straightforward fashion.

Theorem 1. *For $k_1, k_2 \geq 1$, the magnitude of the gradient from a hard positive for TCL loss is* *strictly greater than the magnitude of the gradient from a hard positive for SupCon loss and hence, the following result follows:*

$$\underbrace{|X_{ip} - P^t_{ip} + Y^t_{ip}|}_{\text{(TCL's hard positive gradient)}} > \underbrace{|X_{ip} - P^s_{ip}|}_{\text{(Supcon's hard positive gradient)}} \tag{4.19}$$

Proof. As the authors of [23] show in Section 3 of their supplementary (we also mention the same in Chapter 3) that the gradient from a positive while flowing back through the projector into the encoder reduces to almost zero for easy positives and $|P^s_{ip} - X_{ip}|$ for a hard positive because of the normalization consideration in the projection network combined with the assumption that $z_i.z_p \approx 1$ for easy positives and $z_i.z_p \approx 0$ for hard positives. Proceeding in a similar manner, it is straightforward to see that the gradient response from a hard positive in case of TCL loss is $|P^t_{ip} - X_{ip} - Y^t_{ip}|$. We don't prove this explicitly again since the derivation will be identical to what authors [23] have already shown. One can refer section 3 of the supplementary of [23] for details.

Now, because $k_1, k_2 \geq 1$, it is easy to observe from equations 3.5 and 4.7 that,

$$P^t_{ip} < P^s_{ip} \tag{4.20}$$

And from equation 4.8:

$$Y^t_{ip} > 0 \tag{4.21}$$

Hence, the result follows. This completes the proof. □

Theorem 2. *For fixed k_1, the magnitude of the gradient response from a hard negative for TCL loss — P^t_{in} strictly increases with k_2.*

Proof.

$$P^t_{in} = \frac{k_2 \exp(z_i.z_n/\tau)}{D(z_i)} \tag{4.22}$$

$$= \frac{k_2 \exp(z_i.z_n/\tau)}{\sum_{p' \in P(i)} \exp(z_i.z_{p'}/\tau) + k_1(\sum_{p' \in P(i)} \exp(-z_i.z_{p'})) + k_2(\sum_{n \in N(i)} \exp(z_i.z_n/\tau))} \tag{4.23}$$

$$= \frac{\exp(z_i.z_n/\tau)}{\left(\sum_{p' \in P(i)} \exp(z_i.z_{p'}/\tau) + k_1(\sum_{p' \in P(i)} \exp(-z_i.z_{p'}))\right)/k_2 + (\sum_{n \in N(i)} \exp(z_i.z_n/\tau))} \tag{4.24}$$

It is now easy to observe that for a fixed k_1, P^t_{in} strictly increases with k_2. This completes the proof. □

4.1 Effects of k_1 and k_2

The authors of SupCon loss show (in equation 18 in the supplementary of [23]) that the magnitude of gradient response from a hard positive $|X_{ip} - P^s_{ip}|$ increases with the number of positives and negatives in the batch. This is basically a result of reducing the value of P^s_{ip}, a term that results from having terms denoting the similarity between the anchor and the positives in the denominator of L^{sup}_i. But they approximate the numerator of P^s_{ip} to 1 by assuming $z_i.z_p \approx 0$ for a hard positive which might not always be true (especially since τ is typically chosen to be small like 0.1). As evident from the proof of Theorem 1, we further push this idea and reduce the value of P^s_{ip} in SupCon loss to P^t_{ip} in TCL loss by having an extra term in the denominator involving k_1 — $k_1(\sum_{p' \in P(i)} exp(-z_i.z_{p'}))$ and choosing a large enough value for k_1. Hence, it reduces the effect of implicit consideration of positives as negatives, the first limitation of SupCon loss discussed in chapter 3. Note that having the extra term to increase the gradient response from hard positives is not the same as increasing the gradient response by amplifying the learning rate. This is because for the same and fixed learning rate, TCL loss has higher hard positive gradient magnitude as compared to SupCon loss which is achieved by changing the coefficient of z_p in equation 4.5. This in turn means changing the gradient direction as well. This leads to consistently better performance as shown in the numerous experiments that we perform. Also, it directly follows from Theorem 2 that k_2 allows to regulate (increase) the gradient signal from a hard negative and thus, TCL loss overcomes the second limitation of the SupCon loss.

4.2 Augmentation Strategy for Self-Supervised Setting

Since TCL loss can use multiple positives, we consider working with positive triplets instead of positive pairs in self-supervised settings. Given a batch B with N samples, we produce augmented batch I of size $3N$ by producing three augmented views (positives) for each sample in B. This idea can further be extended in different ways to have more positives per anchor. For example, one can think of combining different augmentation strategies to produce multiple views per sample although we limit ourselves to positive triplets in this work.

Chapter 5

Experiments

We evaluate TCL in three stages: 1. Supervised setting, 2. Hyperparameter stability and 3. Self-supervised setting. We then present empirical analysis on TCL loss's parameters — k_1 and k_2 and show how we choose their values. All the relevant training details are mentioned in the appendix.

5.1 Supervised Setting

We start by evaluating TCL in supervised setting first. Since the authors of [23] mention that SupCon loss performs significantly better than triplet loss [22] and N-pair loss [28], we directly compare TCL loss with SupCon and cross-entropy losses on various classification benchmarks including CIFAR-10, CIFAR-100 [24], Fashion MNIST (FMNIST) [31] and ImageNet-100 [12]. The encoder network chosen is ResNet-50 [21] for CIFAR and FMNIST datasets while Resnet-18 [21] for ImageNet dataset. The representation vector is the activation of the final pooling layer of the encoder. ResNet-18 and ResNet-34 encoders give 512 dimensional representation vectors while ResNet-50 and above produce 2048 dimensional vectors. The projector network is a MLP (multi-layer perceptron) with one hidden layer of size 512 for ResNet-18 and Resnet-34, and size 2048 for ResNet-50 and higher networks. The output layer of the projector MLP is 128 dimensional for all the networks. We use the same cross-entropy implementation as given by Supervised Contrastive Learning [23].

Note that for fair comparison of TCL with Supervised Contrastive Learning, we keep the architecture and all other possible hyperparameters except the learning rate exactly the same. We also do hyperparameter tuning significantly more for Supervised Contrastive Learning than for TCL. As a result, we found that our re-implementation of Supervised Contrastive Learning gave better results than what is reported in the paper [23]. For example, on CIFAR-100 our significantly tuned version of SupCon loss achieves 79.1% top-1 classification accuracy, 2.6% more than what is reported in the SupCon loss paper. As the authors of SupCon loss [23] mention that 200 epochs of contrastive training is sufficient for training a ResNet-50 on complete ImageNet dataset, our observations for the supervised setting case on relatively smaller datasets like CIFAR, FMNIST and ImageNet-100 are consistent with this finding. We train Resnet-50 (and ResNet-18) for a total of 150 epochs – 100 epochs of contrastive training for the encoder and the projector followed by 50 epochs of cross-entropy training for the linear layer. Note that 150 epochs of total training was sufficient for our re-implementation of SupCon loss to achieve better results than reported in the paper (2.6% more on CIFAR-100 and 0.3% more on CIFAR-10). We anyways provide results for 250 epochs of training in the appendix. As Table 5.1 shows, TCL loss consistently performs better than SupCon loss and outperforms cross-entropy loss on all the datasets.

Table 5.1. Comparison of top-1 accuracies of TCL loss with SupCon loss and cross-entropy loss in supervised setting on different datasets. The values in parenthesis for SupCon loss denote the values presented in their paper [23].

Dataset	Cross-Entropy	SupCon	TCL
CIFAR-10	95.0	96.3 (96.0)	96.4
CIFAR-100	75.3	79.1 (76.5)	79.8
FashionMNIST	94.5	95.5	95.7
ImageNet-100	84.2	85.9	86.7

5.2 Hyperparameter Stability

We now show the stability of TCL loss to a range of hyperparameters. We compare TCL loss with SupCon loss on various hyperparameters — encoder architectures, batch sizes, projection network output embedding sizes and different augmentations. For all the hyperparameter experiments we choose CIFAR-100 as the common dataset (unless stated otherwise), set total training epochs to 150 (same as earlier section), temperature τ to 0.1 and use SGD optimizer with momentum=0.9 and weight decay=$1e-4$.

5.2.1 Encoder Architecture

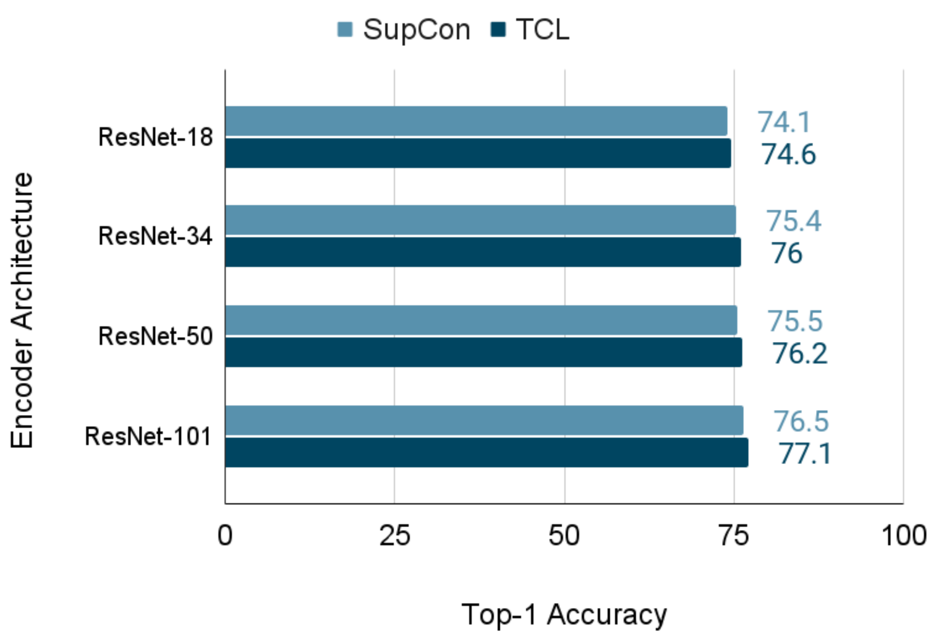

Figure 5.1. SupCon loss vs TCL loss for different encoder architectures.

We choose 4 encoder architectures of varying sizes- ResNet-18, ResNet-34, ResNet-50 and ResNet-101 [21]. For both TCL loss and SupCon loss, we choose batch size as 128 and AutoAugment [10] data augmentation method. As evident from Fig. 5.1, TCL loss achieves consistent improvements in top-1 test classification accuracy over SupCon loss on all the architectures. We also tested TCL loss and SupCon loss on ImageNet-100 with ResNet-18 (batch size of 256) and ResNet-34 (batch size of 128). Using ResNet-18, TCL loss achieved 86.7% top-1 accuracy while SupCon loss achieved 85.9% top-1 accuracy. By switching to ResNet-34, TCL loss got 87.2% top-1 accuracy while SupCon loss got 86.5% top-1 accuracy.

5.2.2 Batch Size

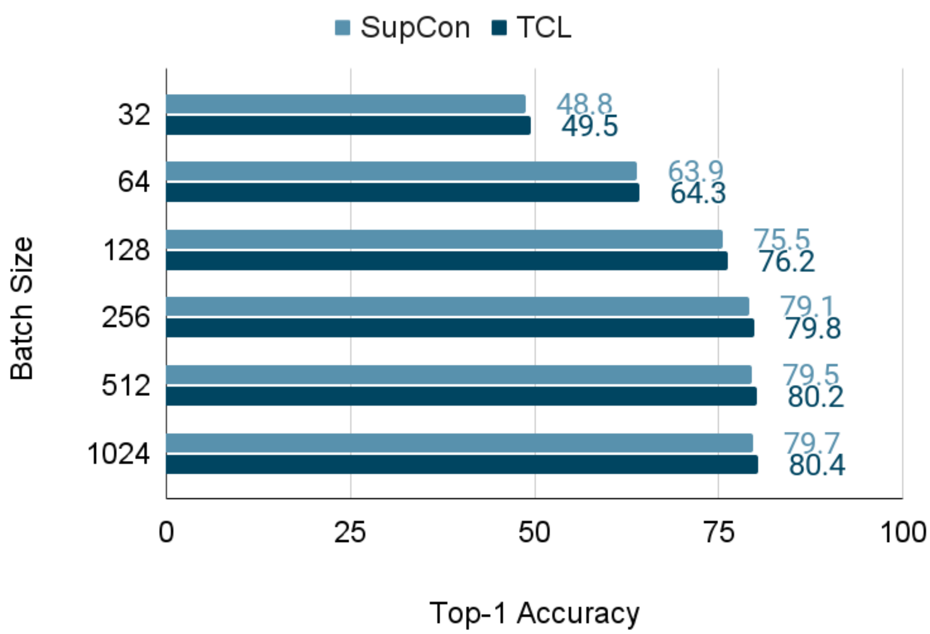

Figure 5.2. SupCon loss vs TCL loss for different batch sizes.

For comparing TCL loss with SupCon loss on different batch sizes, we choose ResNet-50 as the encoder architecture and AutoAugment [10] data augmentation. As evident from Fig. 5.2, we observe that TCL loss consistently performs better than SupCon loss on all batch sizes. All the batch sizes mentioned are after performing augmentation. Note that the authors of SupCon loss use an effective batch size of 256 (after augmentation) for CIFAR datasets in their released code[1]. We select batch sizes equal to, smaller and greater than this value for comparison to demonstrate the effectiveness of Tuned Contrastive Learning.

[1]https://github.com/HobbitLong/SupContrast

5.2.3 Projection Network Embedding (z_i) Size

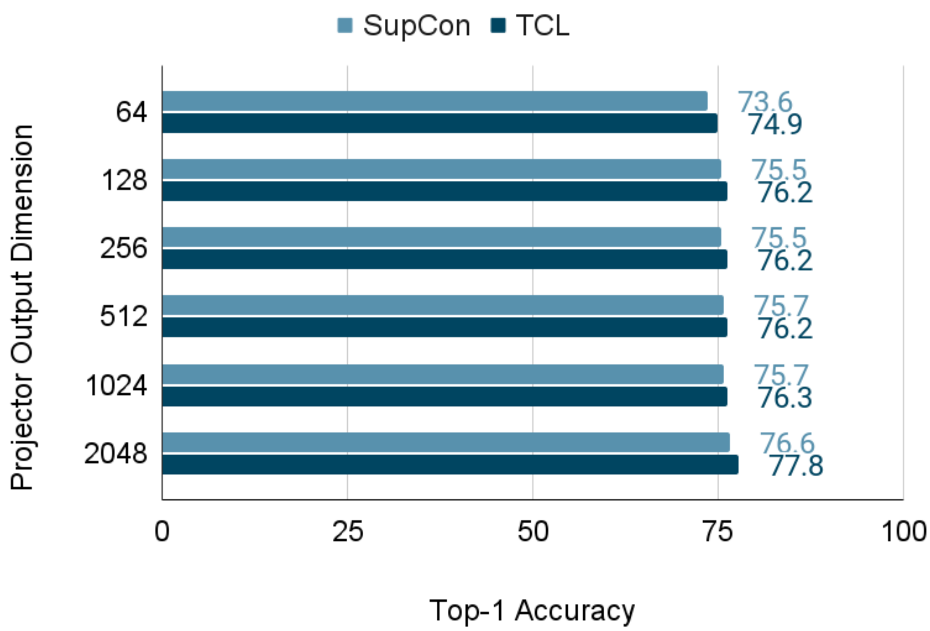

Figure 5.3. SupCon loss vs TCL loss for different projector output sizes.

In this section we analyse empirically how SupCon and TCL losses perform on various projection network output embedding sizes. This particular experiment was not explored as stated by the authors of Supervised Contrastive Learning [23]. ResNet-50 is the common encoder used with Auto-Augment [10] data augmentation. As evident from Fig. 5.3, we observe that TCL loss achieves consistent improvements in top-1 test classification accuracy over SupCon loss for various projector output sizes. We observe that size 64 performs the worst while sizes 128, 256, 512 and 1024 give similar results. Size 2048 performs the best for both with TCL loss achieving 1.2% higher accuracy than SupCon loss on this size.

5.2.4 Augmentations

We choose two augmentation strategies — AutoAugment and SimAugment for comparisons. AutoAugment [10] is a two-stage augmentation policy trained with reinforcement learning and gives stronger (aggressive and diverse) augmentations. SimAugment [6] is rela-

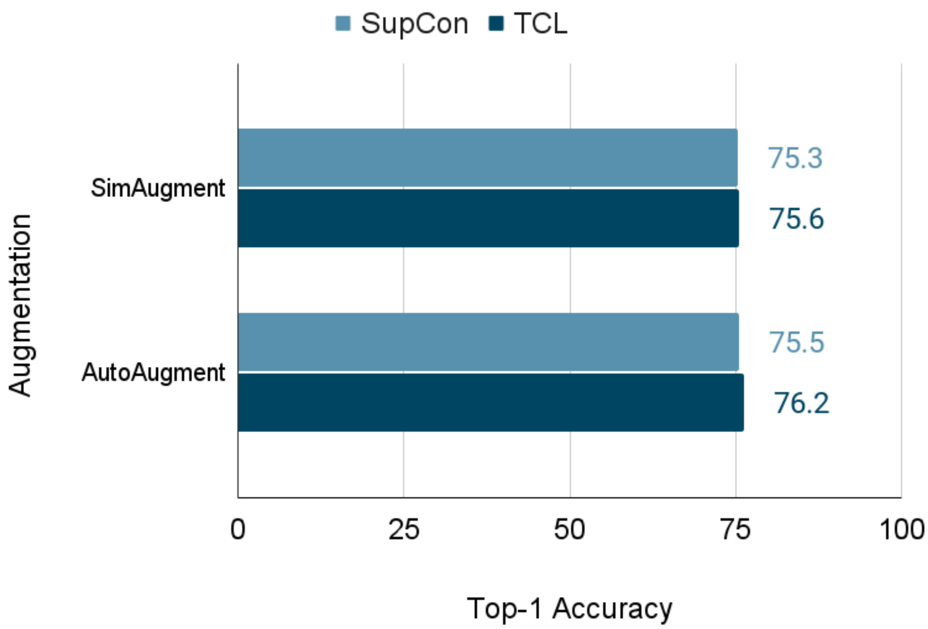

Figure 5.4. SupCon loss vs TCL loss for different augmentation methods.

tively a weaker augmentation strategy used in SimCLR that applies simple transformations like random flips, rotations, color jitters and gaussian blurring. We don't use gaussian blur in our implementation of SimAugment and train for 100 extra epochs i.e. 250 epochs while using it. Fig. 5.4 shows that TCL loss performs better than SupCon loss with both augmentations although, the gain is more with AutoAugment – the stronger augmentation strategy.

5.3 Self-Supervised Setting

In this section we evaluate TCL without any labels in self-supervised setting by making use of positive triplets as described earlier. We compare TCL with various SOTA SSL methods as shown in Table 5.2. The results for these methods are taken from the works of [36], [11]. The datasets used for comparison are CIFAR 10, CIFAR-100 and ImageNet-100. ResNet-18 is the common encoder used for every method. For CIFAR-10 and CIFAR-100 every method uses 1000 epochs of contrastive pre-training including TCL. For ImageNet-100, every method does 400 epochs of contrastive pre-training.

Table 5.2 shows the top-1 accuracy achieved by various methods on the three datasets. TCL performs consistently better than SimCLR [6] and performs on par with various other methods. Note that methods like BYOL [18], VICReg [3], ARB [36] and Barlow-Twins [33] use much larger projector sizes for output embedding and extra hidden layers in the projector MLP to get better performance while MOCO V2 [8] uses a queue size of 32,768 to get better results. Few of the methods like BYOL [18], SimSiam [9], MOCO V2 [20, 8] also maintain two networks and hence, effectively use double the number of parameters.

We also add the results of supervised TCL that can make use of labels as it generalizes to any number of positives. Supervised TCL achieves significantly better results than all other SSL methods. SwAV does use a multi-crop strategy to create multiple augmentations but is not extended to supervised setting to use the labels [4].

Table 5.2. Comparison of top-1 accuracies of TCL with various SSL methods on different datasets. Values in bold show the best performing method for that dataset.

Method	Projector Size	CIFAR-10	CIFAR-100	ImageNet-100
BYOL[18]	4096	92.6	**70.2**	**80.1**
DINO[5]	256	89.2	66.4	74.8
SimSiam[9]	2048	90.5	65.9	77.0
MOCO V2[20, 8]	256	**92.9**	69.5	78.2
ReSSL[37]	256	90.6	65.8	76.6
VICReg[3]	2048	90.1	68.5	79.2
SwAV[4]	256	89.2	64.7	74.3
W-MSE[15]	256	88.2	61.3	69.1
ARB[36]	256	91.8	68.2	74.9
ARB[36]	2048	92.2	69.6	79.5
Barlow-Twins[33]	256	87.4	57.9	67.2
Barlow-Twins[33]	2048	89.6	69.2	78.6
SimCLR[6]	256	90.7	65.5	77.5
TCL (Self-Supervised)	256	91.8	67.2	78.4
TCL (Supervised)	128	95.8	77.5	86.7

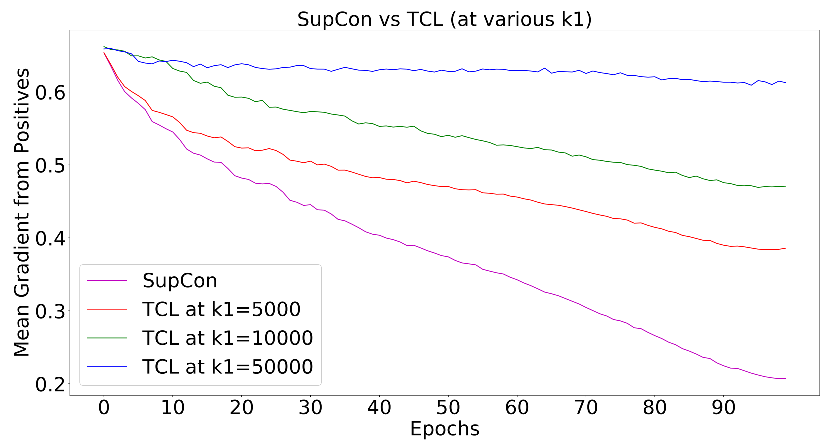

Figure 5.5. Analysis of k_1 — plot of mean gradient magnitude from positives averaged across the batch for SupCon loss and TCL loss (at various values of k_1).

5.4 Analyzing and Choosing k_1 and k_2 for TCL

As we discussed earlier in chapter 4, k_1 helps in increasing the magnitude of gradient from positives while k_2 helps in regulating (increasing) the gradient from negatives. We verify our claims empirically and show how we go about choosing their values for training.

5.4.1 Analyzing effects of k_1

We calculate the mean gradient magnitude from all positives (expressions from equation 4.19) per anchor averaged across the batch and plot the values for SupCon and TCL losses over the course of training of ResNet-50 on CIFAR-100. As evident from Fig. 5.5, increasing the value of k_1 increases the magnitude of gradient response from positives. We also analyze how this correlates with the top-1 accuracy in Fig. 5.6. As we see for small values of k_1, the top-1 accuracy remains more or less the same as that of SupCon loss. As we increase it further, the gradient from positives increase leading to gains in top-1 accuracy. The top-1 accuracy reaches a peak and then starts to drop with further increase in k_1. We hypothesize that this drop is because very large values of k_1 start affecting the gradient response from negatives (equations 4.9 and 4.3). We verify this hypothesis while analyzing k_2.

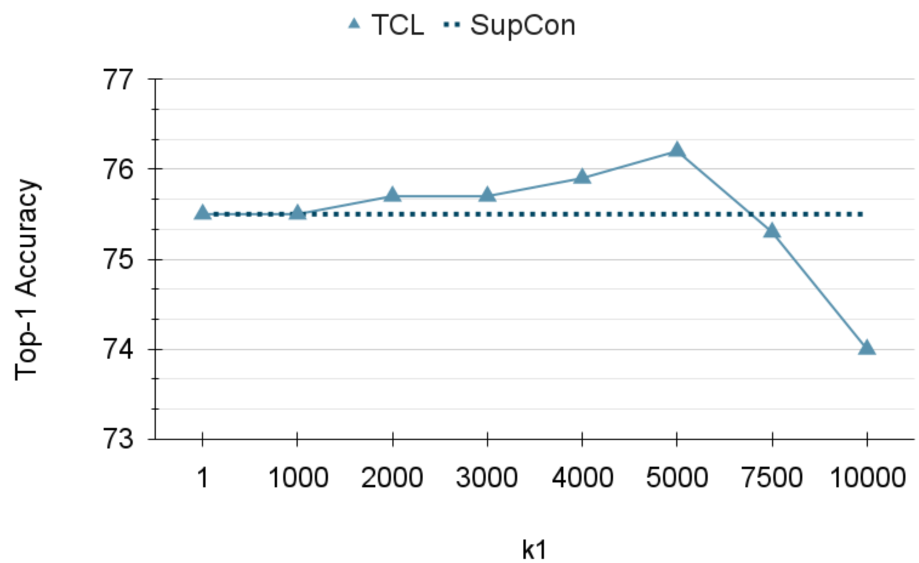

Figure 5.6. Analysis of k_1 — top-1 accuracy vs k_1 on CIFAR-100 for TCL.

5.4.2 Analyzing effects of k_2

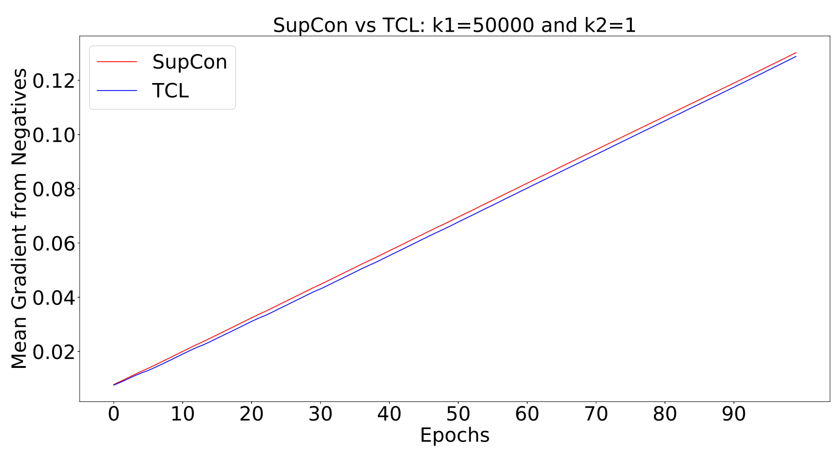

Figure 5.7. Analysis of k_2 — plot of mean gradient magnitude from negatives averaged across the batch for SupCon loss and TCL loss ($k_1 = 50000$ and $k_2 = 1$).

We calculate the mean gradient magnitude from all negatives (expressions from equations 3.6 and 4.9) per anchor averaged across the batch for the same setting as above and plot the values for SupCon and TCL losses. As we see in Fig. 5.7, TCL loss's gradient lags behind SupCon loss's gradient by some margin for $k_1 = 50000$ and $k_2 = 1$. We have chosen a large

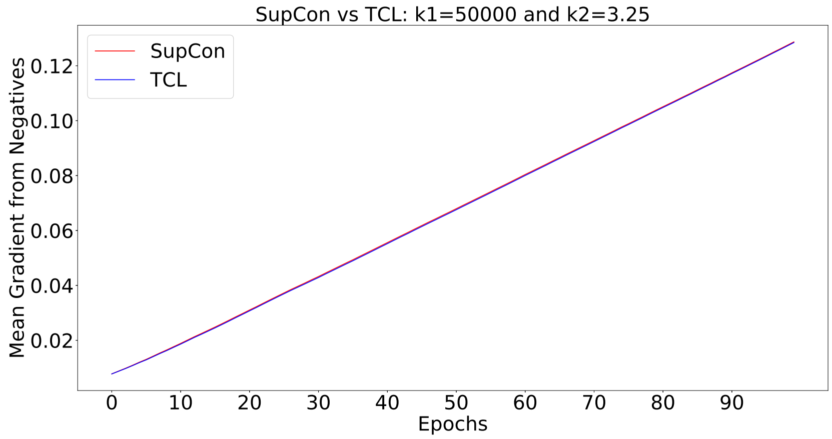

Figure 5.8. Analysis of k_2 — plot of mean gradient magnitude from negatives averaged across the batch for SupCon loss and TCL loss ($k_1 = 50000$ and $k_2 = 3.25$).

value of k_1 here to better show how k_1 affects the gradient response from negatives and how k_2 helps in regulating it. This value of k_1 actually leads to a top-1 accuracy of 71.8%, a drop in performance. When we start increasing the value of k_2, the gradient response from negatives increase for TCL loss. Fig. 5.8 shows that by increasing k_2 to 3.25 while k_1 is fixed at 50000, the gap between gradient (from negatives) curves of TCL loss and SupCon loss vanishes. We also observe that the top-1 accuracy increases back to 76.2%, the best possible accuracy that we got for TCL loss in this setting.

5.4.3 Choosing k_1 and k_2

We observe that a value of k_1 in the range of 10^3 to 10^4 works best with $k_1 = 4 \times 10^3$ or 5×10^3 almost always working on all datasets and configurations we experimented with. We generally start with these two values or otherwise with 2×10^3 and increase it in steps of 2000 till 8×10^3. We also observed during our experiments that choosing any value less than 5×10^3 always gave improvements in performance over SupCon loss. For most of our experiments we set k_1 to 4×10^3 or 5×10^3 and get the desired performance boost in a single run. We found k_2 to be useful to compensate for the reduction in the value of P_{in}^t caused by increasing k_1 and in self-supervised settings where hard negative gradient contribution is important. For setting k_2, we fix k_1 (which itself gives boost in performance) and increase k_2 in steps of 0.1 or 0.2 to see if we can get further improvement. We provide values for k_1 and k_2 for all our experiments in the appendix. As we see, we generally keep $k_2 = 1$ for supervised settings but we do sometimes set it to a value slightly bigger than 1. We set k_2 to a higher value in self-supervised settings as compared to supervised settings to get higher gradient contribution from hard negatives. Increasing k_1 didn't help much in boosting the performance in self-supervised setting (as we only had two positives per anchor) and so we set it to 1. Increasing k_2 also increases the gradient response from positives to some extent by decreasing P_{ip}^t (equation 4.7) and so, we found it sufficient to only increase k_2 and set k_1 to 1 in self-supervised setting.

Chapter 6

Conclusion and Limitations

In this thesis, we have presented a novel contrastive loss function called Tuned Contrastive Learning (TCL) loss that generalizes to multiple positives and multiple negatives present in a batch and is applicable to both supervised and self-supervised settings. We showed mathematically how its gradient response to hard positives and hard negatives is better than that of SupCon loss. We evaluated TCL loss in supervised and self-supervised settings and showed that it performs on par with existing state-of-the-art supervised and self-supervised learning methods. We also showed empirically the stability of TCL loss to a range of hyperparameter settings.

A limitation of our work is that the proposed loss objective introduces two extra parameters k_1 and k_2, for which the values need to be set heuristically. Future direction can include works that come up with loss objectives that provide the properties of TCL loss out of the box without introducing any extra parameters.

Chapter 7

Broader Impact

We hope that the work presented in this thesis inspire people to consider contrastive learning as an alternative to cross-entropy loss based learning which is commonly used in many fields such as computer vision and natural language processing. Unlike cross-entropy loss, contrastive learning can be used for unsupervised learning as well and allows to learn from unlabelled data, that is often easily accessible, for downstream tasks. We will release our code publicly for the benefit of the research community.

This thesis, in its entirety, has been submitted for publication with the possibility of its material being included in a conference in 2023, authored by Chaitanya Animesh and Manmohan Chandraker. The thesis author was the primary investigator and the first author of this paper.

Appendix A

Training Details

A.1 Supervised Setting

We first present the common training details used for each dataset experiment in the supervised setting for Supervised Contrastive Learning [23] and TCL. Except for the contrastive training learning rate, every other detail presented is common for SupCon loss and TCL loss. As mentioned in chapter 5, we train for a total of 150 epochs which involves 100 epochs of contrastive training for the encoder and the projector, and 50 epochs of cross-entropy training for the linear layer for both the losses. AutoAugment [10] is the common data augmentation method used except for FMNIST [31] for which we used a simple augmentation strategy consisting of random cropping and horizontal flip. We use cosine annealing based learning rate scheduler and SGD optimizer with momentum=0.9 and weight decay=$1e-4$ for both contrastive and linear layer training. Temperature τ is set to 0.1. For linear layer training, the starting learning rate is $5e-1$. ResNet-50 [21] is the common encoder architecture used.

A.1.1 CIFAR-10

Images are resized to 32×32 in the data augmentation pipeline. We use a batch size of 128. For both SupCon and TCL losses we use a starting learning rate of $1e-1$ for contrastive training. We set $k_1 = 5000$ and $k_2 = 1$ for TCL.

A.1.2 CIFAR-100

Images are resized to 32×32 in the data augmentation pipeline. We use a batch size of 256. For both SupCon and TCL losses we use a starting learning rate of $2e-1$ for contrastive training. We set $k_1 = 4000$ and $k_2 = 1$ for TCL.

A.1.3 FMNIST

Images are resized to 28×28 in the data augmentation pipeline. We use a batch size of 128. For both SupCon and TCL losses we use a starting learning rate of $9e-2$ for contrastive training. We set $k_1 = 5000$ and $k_2 = 1$ for TCL.

A.1.4 ImageNet-100

Images are resized to 224×224 in the data-augmentation pipeline and batch size of 256 is used. For SupCon loss we use a starting learning rate of $2e-1$ for contrastive training while $3e-1$ for TCL loss. We set $k_1 = 4000$ and $k_2 = 1$ for TCL loss.

A.1.5 More Experiments

Table A.1. Comparison of top-1 accuracies of TCL with Supervised Contrastive Learning in supervised setting on different datasets for 250 epochs of training.

Dataset	SupCon	TCL
CIFAR-10	96.7	96.8
CIFAR-100	81.0	81.6
FashionMNIST	95.5	95.7
ImageNet-100	86.5	87.1

For CIFAR-100 dataset and batch size of 128, we also ran comparison experiment 30 times to get 95% confidence intervals for top-1 accuracies of SupCon and TCL losses. For SupCon loss we got $74.79 \pm 0.23\%$ as the confidence interval while for TCL loss we got $75.72 \pm 0.16\%$ as the confidence interval. We also present results for 250 epochs of training constituted by 200 epochs of contrastive training and 50 epochs of linear layer training in Table

A.1. As we see, TCL loss performs consistently better than SupCon [23] loss. Note that we didn't see any performance improvement for FMNIST dataset for either SupCon loss or TCL loss by training for 250 epochs.

A.2 Hyperparameter Stability

For the hyperparameter stability experiments we have presented most of the details in chapter 5. We present the learning rates and values of k_1 and k_2 used for TCL. Remaining details are the same as the supervised setting experiments.

A.2.1 Encoder Architecture

The starting learning rate for contrastive training is $1e-1$ for all the encoders except ResNet-101 for which we used a value of $9e-2$. $k_1 = 5000$ and $k_2 = 1$ are the common values used for all the encoders.

A.2.2 Batch Size

For batch sizes=32, 64, 128, 256, 512 and 1024 we set the starting learning rates for contrastive training to $8e-3$, $9e-3$, $1e-1$, $2e-1$, $5e-1$ and 1 respectively. For batch size of 32 we used $k_1 = 5000$ and $k_2 = 1$. For batch size of 64 we used $k_1 = 7500$ and $k_2 = 1$. For batch size of 128 we used $k_1 = 5000$ and $k_2 = 1$. For batch sizes of 256, 512 and 1024 we used $k_1 = 4000$ and $k_2 = 1$.

A.2.3 Projection Network Embedding (z_i) Size

We used a common starting learning rate of $1e-1$ with $k_1 = 5000$ and $k_2 = 1$ for all the projector output sizes for contrastive training.

A.2.4 Augmentations

For AutoAugment [10] method, we use a learning rate of $1e-1$ with $k_1 = 5000$ and $k_2 = 1$ for contrastive training. For SimAugment [6], we use a learning rate of $1e-1$ with $k_1 = 5000$ and $k_2 = 1.2$ for contrastive training.

A.3 Self-Supervised Setting

For the self-supervised setting, we reuse the code provided by [11] and we are thankful to them for providing all the required details. The projector used for TCL is exactly the same as SimCLR in [11], [36] for fair comparison and consists of one hidden layer of size 2048 (4096 for ImageNet-100) and output size of 256. ResNet-18 is the common encoder used for all the methods. Label information is not used for contrastive pre-training. We use cosine annealing based learning rate scheduler and SGD optimizer with momentum=0.9 wrapped with LARS optimizer [32] and weight decay of $1e-4$ for contrastive pre-training. Augmentation used is SimAugment [6] and is done in the same manner as [11]. Gaussian blur is used in the data augmentation pipeline for self-supervised setting. The linear layer is trained for evaluation using the cross-entropy loss and the labels for 100 epochs with cosine annealing based learning rate scheduler and starting learning rate of $5e-1$ for all the datasets.

A.3.1 CIFAR-10

All methods do 1000 epochs of contrastive pre-training on CIFAR-10 [24] and images are rescaled to 32×32 in the data augmentation pipeline. We set batch size=256, same as SimCLR. For TCL, we use a starting learning rate of $5e-1$ for contrastive pre-training with $k_1 = 1$ and $k_2 = 1.5$.

A.3.2 CIFAR-100

All methods do 1000 epochs of contrastive pre-training on CIFAR-100 [24] and images are rescaled to 32×32 in the data augmentation pipeline. We set batch size=256, same as

SimCLR. For TCL, we use a starting learning rate of $5e-1$ for contrastive pre-training with $k_1 = 1$ and $k_2 = 1.5$.

A.3.3 ImageNet-100

All methods do 400 epochs of contrastive pre-training on ImageNet-100 [12] and images are rescaled to a size of 224×224. We set batch size=256, same as SimCLR. For TCL, we use a starting learning rate of $5e-1$ for contrastive pre-training with $k_1 = 1$ and $k_2 = 1.5$. We use the same subset of 100 ImageNet classes as used by SimCLR [6] and all the other methods in [36].

Bibliography

[1] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[2] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, Avi Schwarzschild, Andrew Gordon Wilson, Jonas Geiping, Quentin Garrido, Pierre Fernandez, Amir Bar, Hamed Pirsiavash, Yann LeCun, and Micah Goldblum. A cookbook of self-supervised learning, 2023.

[3] Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022.

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 13–18 Jul 2020.

[7] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 403–412, 2017.

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning, 2020.

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

[10] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[11] Victor Guilherme Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. solo-learn: A library of self-supervised methods for visual representation learning. *Journal of Machine Learning Research*, 23(56):1–6, 2022.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[13] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[14] Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. Large margin deep networks for classification. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[15] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. *CoRR*, abs/2007.06346, 2020.

[16] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[18] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020.

[19] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, 2006.

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *Similarity-Based Pattern Recognition: Third International Workshop, SIMBAD 2015, Copenhagen, Denmark, October 12-14, 2015. Proceedings 3*, pages 84–92. Springer, 2015.

[23] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc., 2020.

[24] A. Krizhevsky and G Hinton. Learning multiple layers of features from tiny images, 2009.

[25] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks, 2017.

[26] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft, 2013.

[27] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016.

[28] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[29] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels, 2015.

[30] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.

[31] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[32] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[33] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

[34] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 649–666. Springer, 2016.

[35] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1058–1067, 2017.

[36] Shaofeng Zhang, Lyn Qiu, Feng Zhu, Junchi Yan, Hengrui Zhang, Rui Zhao, Hongyang Li, and Xiaokang Yang. Align representations with base: A new approach to self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16600–16609, June 2022.

[37] Mingkai Zheng, Shan You, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Ressl: Relational self-supervised learning with weak augmentation. *Advances in Neural Information Processing Systems*, 34:2543–2555, 2021.