

Dual Temperature Helps Contrastive Learning Without Many Negative Samples: Towards Understanding and Simplifying MoCo

Chaoning Zhang^{1,*}, Kang Zhang^{1,*}, Trung X. Pham^{1,*}, Axi Niu², Zhinan Qiao³

Chang D. Yoo¹, In So Kweon¹

¹ KAIST, ² Northwestern Polytechnical University, ³ University of North Texas

Abstract

Contrastive learning (CL) is widely known to require many negative samples, 65536 in MoCo for instance, for which the performance of a dictionary-free framework is often inferior because the negative sample size (NSS) is limited by its mini-batch size (MBS). To decouple the NSS from the MBS, a dynamic dictionary has been adopted in a large volume of CL frameworks, among which arguably the most popular one is MoCo family. In essence, MoCo adopts a momentum-based queue dictionary, for which we perform a fine-grained analysis of its size and consistency. We point out that InfoNCE loss used in MoCo implicitly attract anchors to their corresponding positive sample with various strength of penalties and identify such inter-anchor hardness-awareness property as a major reason for the necessity of a large dictionary. Our findings motivate us to simplify MoCo v2 via the removal of its dictionary as well as momentum. Based on an InfoNCE with the proposed dual temperature, our simplified frameworks, Sim-MoCo and SimCo, outperform MoCo v2 by a visible margin. Moreover, our work bridges the gap between CL and non-CL frameworks, contributing to a more unified understanding of these two mainstream frameworks in SSL. Code is available at: <https://bit.ly/3LkQbaT>.

1. Introduction

Self-supervised learning (SSL) has become increasingly popular in various domains, ranging from NLP [12, 30, 35, 39, 43] to visual representation [6, 22, 37], in the past few years. Especially, contrastive learning (CL) frameworks [2, 6, 22, 23, 25, 37, 45, 48, 51, 54, 66] have attracted significant attention due to its intuitive motivation. In essence, CL is designed to attract the anchor sample [47] close to the positive sample, *i.e.* another augmented view of the same image, and simultaneously repulse it from negative

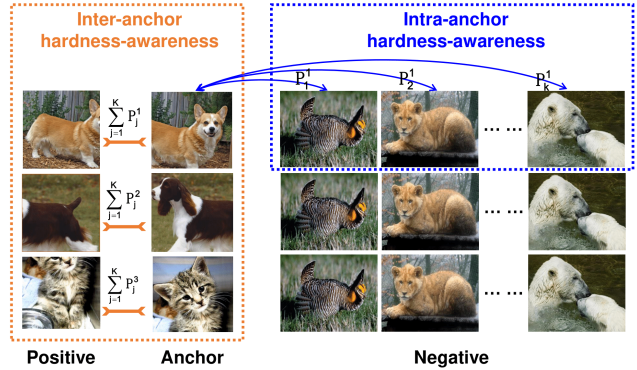


Figure 1. Intra-anchor and Inter-anchor hardness-aware properties. The former is indicated by different repulsing weights (see Eq 5) for different negative samples based on their hardness, *i.e.* $p_1^1 \neq p_2^1 \dots \neq p_K^1$ (K denotes NSS), and the latter is indicated by different weights being put on different anchor samples to attract the corresponding positive sample, *i.e.* $\sum_{j=1}^K p_j^1 \neq \sum_{j=1}^K p_j^2 \neq \sum_{j=1}^K p_j^3$ with three anchor images as a motivation example.

samples, *i.e.* views from different images. With the popular InfoNCE loss [37], CL is widely reported to require a large amount of negative samples [22]. For example, [6] shows that increasing the mini-batch size (MBS) to a large value, 4096 for instance, is essential for achieving competitive performance, for which there are multiple challenges, such as GPU memory concern or difficulty to train with a large MBS [6, 58]. Thus, a major line of CL frameworks, such as MoCo [22], have emerged to decouple the required large negative sample size (NSS) from the MBS with a dynamic dictionary. Despite much effort in the dictionary design [22, 54], why contrastive InfoNCE requires a large dictionary (or many negative samples) is not well understood.

Our investigation of the above problem centers around an interesting hardness-aware property [47] of InfoNCE. A large volume of works [3, 26, 28, 36, 46, 61] have studied strategies of mining hard negative samples, *i.e.* those samples that are similar to the anchor sample. We point out that the anchor sample also has this hardness property. Concep-

*equal contribution. corresponding author: Chaoning Zhang chaoningzhang1990@gmail.com

tually, an anchor sample is considered hard when it is still far from the positive sample and/or close to negative samples. InfoNCE loss has been identified to have the hardness-aware property [47], which contributes to dimensional de-correlation [62], is critical for performance.

Prior works [47, 62] mainly study the hardness-awareness *within an anchor*, which is therefore termed *intra-anchor* hardness-aware property here. As in Figure 1, it indicates that the gradient puts different weights (see Eq 5) on various negative samples for repulsing the anchor from them with different strength of penalties, *i.e.* $p_1^i \neq p_2^i \dots \neq p_K^i$ (the fixed superscript i , 1 for instance, denotes the same anchor). In contrast, the *inter-anchor* hardness-aware property indicates different weights being on anchors for attracting them to their corresponding positive sample with different penalties, *i.e.* $\sum_{j=1}^K p_j^1 \neq \sum_{j=1}^K p_j^2 \neq \sum_{j=1}^K p_j^3$ with three anchors as an example.

Overall, our contributions are summarized as follows.

- We point out that anchors have hardness property, for which contrastive InfoNCE loss by default attracts them to their corresponding positive samples with various strength of penalties. Recognizing this, we disentangle InfoNCE into vector and scalar components that reflect intra-anchor and inter-anchor hardness-aware properties, respectively. Such a decomposed loss facilitates a fine-grained analysis on the MoCo dictionary and we reveal: (i) a small dictionary is sufficient for the vector component which, however, requires high consistency between encoders for representing the negative and positive keys; (ii) the scalar component requires a very large dictionary but is less sensitive to such consistency.
- We identify that the increase of dictionary size and temperature both help alleviate the inter-anchor hardness-aware sensitivity for improving performance. Our findings help simplify MoCo family via removing their dictionary and momentum. Specifically, we propose dual temperature for realizing independent control of intra-anchor and inter-anchor properties. Without a dictionary, our proposed SimMoCo achieves comparable or superior performance over the baseline MoCo v2. Notably, our dictionary-free and momentum-free SimCo is simple yet effective.
- Our investigation helps bridge the gap between CL and non-CL frameworks, contributing to a unified perspective on these two major SSL frameworks.

2. Related Work

Recently, multiple works [4, 8, 15, 19, 60] have attempted SSL without using negative samples, demonstrative performance comparable to the CL frameworks. However, they

are often dependent on additional predictor [19] or stop gradient [8] or explicit de-centering and de-correlation [4, 15, 60, 62]. CL remains as a mainstream framework for SSL and has also been extensively studied in other filed applications [1, 13, 16, 27, 38, 47, 52, 55, 56, 59, 65].

Contrastive learning. The core of unsupervised learning is to learn augmentation-invariant representation, for which CL is at the core of its development [17, 34, 41, 42, 50]. Inspired by this success, CL has been extensively studied for SSL pretext training [2, 6, 23, 25, 37, 44, 54, 66]. Early works have attempted margin-based contrastive losses [21, 24, 50] and [37, 54] propose a NCE-like loss which has become the de facto standard loss in CL.

More recently, demonstrating superior performance over supervised pre-training counterparts, MoCo [22] has attracted significant attention. MoCo v2 [7] incorporates stronger augmentation and additional MLP projector head from [6], which shows significant performance improvement over MoCo v1. Moreover, [9] has demonstrated that MoCo family can also exploit ViT structures [14] based on which they find that prediction head from the non-CL frameworks [8, 19] brings additional performance boost. In essence, what is unique to MoCo family is their dictionary, where the keys are also found to benefit from increased diversity through negative interpolation [64]. The understanding of this core component, *i.e.* momentum-based queue dictionary, in MoCo is limited and our work fills the gap to perform a fine-grained analysis.

A key property of CL is that it involves negative samples, and a major line works [3, 10, 26, 28, 29, 36, 40, 46, 49, 53, 61] have shown that mining hard negative samples can be beneficial for performance. Moreover, [47] has identified that InfoNCE has a hardness-aware property which is critical for competitive performance. [62] has shown that this can be attributed to the effect of dimensional de-correlation. In contrast to them that mainly focused on intra-anchor hardness-awareness, our work studies the inter-anchor hardness-aware property and identifies it as a major reason to explain why MoCo family requires a large dictionary.

Temperature in CL.

Temperature plays a key role for the success of CL due to its *hardness-aware* [47] or de-correlation [62]. They analyze the influence of the temperature in the vector component, while ours is the first to decompose the influence of temperature into two components. The dual temperature has been previously studied in a non-CL framework termed DINO [5] from the perspective of knowledge distillation. Specifically, the teacher adopts a lower temperature than that of the student for help distilling knowledge. By contrast, our work adopts dual temperature in a contrastive InfoNCE for realizing independent control of two hardness-aware sensitiveness. Recently, [63] has also exploited input-dependent learnable temperature in SSL for estimating un-

certainty in out-of-distribution detection. It might be interesting to apply our dual temperature concept to [63] for identifying which one (or both) is beneficial for such uncertainty estimation.

3. Background

A large dictionary is desirable. Driven by various motivations, multiple works [2, 23, 25, 37, 44, 54, 66] have designed dynamic dictionaries and exploited the stored keys as negative samples. This dictionary is desirable to be large, for which [54] proposes to save the representations of all training samples in a memory bank. To increase the consistency among the stored representations, MoCo [22] proposes a FIFO queue dictionary based on the momentum encoder. The influence of such consistency on MoCo is demonstrated in [64] by analyzing the effect of momentum coefficient. Without a dictionary, the negative sample size would be limited by the MBS. The main merit of a dictionary is to decouple the NSS from the MBS, which allows access to a large number of negative samples without increasing the MBS. Despite many attempts at exploring various dictionaries, less attention has been paid to understanding why CL requires a large dictionary.

Contrastive loss. NCE-like loss [20] has been independently introduced with various motivations in multiple popular works [37, 42, 54] and it has emerged as the de-facto standard loss for CL. Following [22, 37, 62], we term it InfoNCE for consistency. Given an encoder f , a random input sample x is encoded as a query (or anchor) q or key k , which are often l_2 normalized to avoid scale ambiguity. We consider a set of encoded queries (anchors) $\{q_1, q_2, \dots\}$ and encoded keys $\{k_1, k_2, \dots\}$. With similarity measure by dot product, InfoNCE is formulated as:

$$\mathcal{L}_{q_i} = -\log \frac{\exp(q_i \cdot k_+ / \tau)}{\exp(q_i \cdot k_+ / \tau) + \sum_{j=1}^K \exp(q_i \cdot k_j / \tau)} \quad (1)$$

where k_+ denotes the positive key to anchor q_i and τ denotes the temperature. This loss has low value when q_i is similar to its positive key and dissimilar to negative keys.

Hardness-aware property. To guide an anchor close to its positive key and far from negative keys, a simple loss as

$$\mathcal{L}_{simple} = -q_i \cdot k_+ + \frac{1}{K} \sum_{j=1}^K q_i \cdot k_j, \quad (2)$$

has been designed in [47]. The gradient on q_i is derived as

$$\frac{\partial \mathcal{L}_{simple}}{\partial q_i} = -(k_+ - \frac{1}{K} \sum_{j=1}^K k_j), \quad (3)$$

which shows that the penalty weight on each negative key is the same. [47] has identified that InfoNCE outperforms

the above simple loss due to its hardness-aware property via putting more penalty weight on those hard keys. This is reflected in the derived gradient of Eq 1 on q as

$$\frac{\partial \mathcal{L}_{q_i}}{\partial q_i} = - \left(\left(\sum_{j=1}^K p_j^i \right) k_+ - \sum_{j=1}^K p_j^i k_j \right), \quad (4)$$

where a constant component $\frac{1}{\tau}$ is omitted for simplicity because it can be perceived as part of the learning rate. p_j^i conceptually indicates the probability of q_i being recognized as k_j , which is detailed as

$$p_j^i = \frac{\exp(q_i \cdot k_j / \tau)}{\exp(q_i \cdot k_+ / \tau) + \sum_{r=1}^K \exp(q_i \cdot k_r / \tau)}. \quad (5)$$

Note that for a fixed query q_i , p_j^i ($j \in [1, K]$) are those weights in the intra-anchor hardness-awareness of Fig 1. Proportional to $\exp(q_i \cdot k_j / \tau)$, p_j^i indicates more penalty weight being put on hard negative samples [47].

4. Towards Understanding MoCo

The core of the seminal MoCo centers around its momentum encoder (MoEn)-based dictionary [22]. Our work revisits a prior hypothesis and performs a fine-grained analysis on its dictionary for a new understanding of MoCo.

Prior hypothesis. It is hypothesized that the dictionary needs to be *large* and *consistent* [22]. Regarding the size, it is assumed in [22] that “Intuitively, a larger dictionary may better sample the underlying continuous, high-dimensional visual space”. Straightforwardly, [22] attributes the necessity of a large dictionary size to the requirement of sampling. Indeed, as they claim, in general, it is *intuitive* that more samples are necessary for better modelling a more high-dimensional space. However, whether this is indeed the major reason for the requirement of a large dictionary remains unclear. On the other hand, regarding the consistency, it is argued in [22] that “the keys in the dictionary should be represented by the same or similar encoder so that their comparisons to the query are consistent”. The stored keys in the dictionary from the past iterations are used as negative keys, thus the necessity of MoEn was mainly attributed to such negative-negative (NN) consistency. Here, we attempt to examine the above claims with a focus on InfoNCE’s hardness-aware property.

4.1. Inter-Anchor Hardness-Aware Property

It has been noted in [47] that in the gradient on the anchor, the weight on the positive key is equal to the sum of weights on all negative keys, *i.e.* $\sum_{j=1}^K p_j^i$ (see Eq 4). Prior work is mainly interested in the unevenness of those p_j^i within an anchor. Our work pays attention to the value

of this sum and it motivates us to decompose Eq 4 as

$$\frac{\partial \mathcal{L}_{q_i}}{\partial q_i} = - \underbrace{\sum_{j=1}^K \mathbf{p}_j^i}_{\text{Inter-anchor hardness-awareness}} \left(k_+ - \underbrace{\sum_{j=1}^K \hat{\mathbf{p}}_j^i}_{\text{Intra-anchor hardness-awareness}} k_j \right), \quad (6)$$

where $\hat{\mathbf{p}}_j^i = \mathbf{p}_j^i / \sum_{j=1}^K \mathbf{p}_j^i$. With such decomposition, we note that the weight on positive key is still equal to the sum of $\hat{\mathbf{p}}_j^i$ ($1 = \sum_{j=1}^K \hat{\mathbf{p}}_j^i$). Clearly, $\sum_{j=1}^K \mathbf{p}_j^i$ is an anchor-wise weight for indicating the hardness of anchor q_i (see Fig. 1).

Loss transformation. We reformulate Eq 1 as:

$$\mathcal{L}_{q_i}^{new} = \text{sg}\left[\frac{\partial \mathcal{L}_{q_i}}{\partial q_i}\right] \cdot q_i = \underbrace{\text{sg}\left[\sum_{j=1}^K \mathbf{p}_j^i\right]}_{\text{Scalar component}} q_i \cdot \underbrace{\text{sg}\left[\left(k_+ - \sum_{j=1}^K \hat{\mathbf{p}}_j^i k_j\right)\right]}_{\text{Vector component}}, \quad (7)$$

where $\text{sg}[\cdot]$ indicates the *stop gradient*. Intra-anchor and inter-anchor hardness-awareness are reflected in the vector and scalar components, respectively. This loss transformation enables the adoption of independent dictionaries for the two components. The above loss is mathematically equivalent to that in Eq 1 for optimizing q because they share the same gradient on q (omitted constant $\frac{1}{\tau}$ is considered in practical implementation). Note that k in Eq 1 has no gradients because they are from the dictionary or the output of the momentum encoder [22].

The above gradient decomposition and loss transformation facilitate the analysis of the dictionary. Our following analysis is based on MoCo v2, however, for concept and notation simplicity, it is still referred to as MoCo.

4.2. A Fine-Grained Analysis on MoCo Dictionary

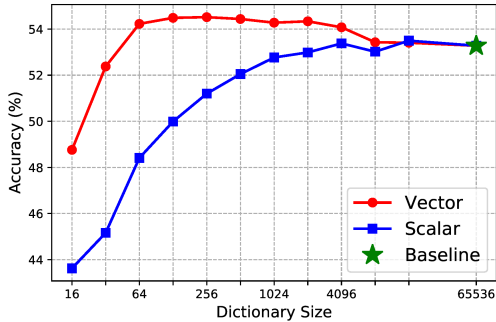


Figure 2. Influence of the size of a independent dictionary. The star indicates the baseline setting the size of both D_{vector} and D_{scalar} to 65536. Fixing the D_{scalar} to 65536, the red line shows the influence of D_{vector} size. Fixing the D_{vector} to 65536, the blue line shows the influence of D_{scalar} size. The experiments are performed on CIFAR100 with MoCo v2 for 200 epochs. Detailed setup is included in the supplementary.

Size. By default, MoCo adopts a very large dictionary size, 65536 for instance, and we treat it as the baseline of our investigation. Since CL requires a large dictionary size, decreasing the size is confirmed to decrease the performance. The dictionary size has an influence on both vector and scalar components. To disentangle such influence, based on the loss in Eq 7, we adopt two independent dictionaries, D_{scalar} and D_{vector} , for the scalar and scalar components, respectively. We investigate two scenarios: (a) adopting various D_{scalar} sizes with the D_{vector} size fixed to 65536; and (b) adopting various D_{vector} sizes with the D_{scalar} size fixed to 65536.

From the results in Figure 2, there are two major observations. First, the scalar component is highly sensitive to its dictionary size and the performance is much worse when D_{scalar} is small. Second, a dictionary size as small as 64 in the vector component is already sufficient for competitive performance. Interestingly, for the vector component, the performance is optimal when the dictionary size is around 256, *i.e.* only the keys stored in the last iteration is used since the MBS is set to 256 in this setup.

Sampling strategies	Earliest	Random	Newest
Top-1 Accuracy(%)	1	44.03	53.43

Table 1. Comparison of various sampling strategies on CIFAR100

Consistency. [22] mainly attributes the quality of stored keys to NN consistency. By contrast, we conjecture that it might be more important for the positive and negative keys to be represented by the same or similar encoders. Since the keys from the current MoEn are used as the positive one, it is straightforward that the stored order might be an important factor with positive-negative (PN) consistency considered.

To verify our conjecture, with a full D_{scalar} used, we sample K_{vector} keys from the D_{vector} . The investigated sampling strategies are as follows: (a) sampling the earliest K_{vector} keys; (c) randomly sampling K_{vector} keys; (c) sampling the most recent K_{vector} keys. Recall that adopting 256 keys for the vector component actually outperforms that with a very large dictionary, and we set K_{vector} to 256. Sampling the earliest K_{vector} keys guarantees high NN consistency, however, the results in Table 1 show that it leads to non-convergence. With such a sampling strategy, we confirm that increasing K_{vector} to a much larger value, 4096 for instance, does not alleviate such collapse. On the other hand, random sampling has very low NN consistency but high PN consistency leads to a reasonable performance but under-performs that with the most recent keys. Overall, the results suggest that (a) PN consistency better indicates the quality of the stored keys; (b) the vector component is sensitive to such consistency. This also helps explain the in-

interesting phenomenon in Figure 2 that adopting keys only from the last iteration outperforms that with a very large dictionary. For the vector component, *a large dictionary is not always optimal from the perspective of PN consistency since it contains many old keys*. By contrast, a larger dictionary in the scalar component consistently improves the performance (see Figure 2), suggesting the scalar component is less sensitive to the quality of the keys. A more detailed discussion on this is in the supplementary.

Relation to prior hypothesis. (i) Regarding the size, it might be tempting to believe that the vector component, which matches the high-dimensional representation space, is more sensitive to the dictionary size. Our results show that a relatively small dictionary size is sufficient for the vector component. (ii) Regarding the consistency, as discussed above, our work shows that their suggested NN consistency [22] is less informative than PN consistency for indicating the quality of the keys. The importance of PN consistency also helps justify their FIFO queue strategy [22].

5. Towards Simplifying MoCo

The dictionary requires additional memory to store negative keys and the keys need to be encoded by a *momentum encoder* to increase their consistency (or quality) in MoCo [22]. Such dictionary and momentum increase the framework complexity, which motivates us to check the possibility to remove them without performance drop. Somewhat surprisingly, our proposed simplified frameworks actually achieve superior performance over the baseline MoCo v2. The simplifying procedure, as well as the underlying rationale are detailed in the following.

5.1. Dictionary Removal

For investigating the role of temperature in controlling the strength penalties on negative sample, [47] defines $r_i(s_{i,j}) = \mathbf{p}_j^i / \sum_{j=1}^K \mathbf{p}_j^i$, i.e. $\hat{\mathbf{p}}_j^i$, as the relative penalty on negative key k_j for the anchor q_i and analyzes its entropy. Inspired by it, we define r_+^i as the relative penalty weight on anchor q_i to attract their corresponding positive sample:

$$r_+^i = \sum_{j=1}^K \mathbf{p}_j^i / \sum_{i=1}^N \sum_{j=1}^K \mathbf{p}_j^i. \quad (8)$$

From the results in Figure 3, we observe that the entropy of r_+ consistently decreases as the dictionary size decreases. On the other hand, its entropy also decreases when the temperature is set lower. A lower entropy indicates the relative penalty weight on each anchor is less equal, i.e. more relative weight on the hard anchors. This mirrors the finding in [47] that a low temperature decreases the entropy of the $r_i(s_{i,j})$ causing more penalty on the hard negative samples. From this observation, we conjecture that the performance with a small dictionary in the scalar component

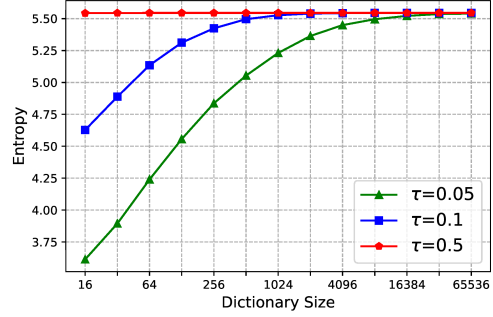


Figure 3. Entropy of r_+ under various dictionary sizes and temperature τ values.

might benefit from a larger temperature similar to the performance boost caused by a larger dictionary size. Temperature in the vector and scalar components are denoted by τ_α and τ_β , respectively. To exclude the influence of temperature change in the vector component, we keep $\tau_\alpha = 0.1$ fixed and only change the value of τ_β . The dictionary size is set to 256 for both scalar and vector components. The results are shown in the first row of Table 2.

Key type	Scalar-temperature (τ_β)				
	0.05	0.1	0.5	1.0	5
Last	43.56	49.74	54.07	54.03	53.89
Current	44.19	49.89	54.45	54.11	54.0

Table 2. Influence of scalar temperature τ_β on CIFAR100. $\tau_\alpha = 0.1$ in all experiments here.

SimMoCo. We observe that a low temperature, 0.05 for instance, leads to a significant performance drop (43.56%), while a sufficiently large temperature is beneficial for superior performance. Note that by default MoCo does not use the current mini-batch keys as the negative samples. Thus the above-discussed results are based on the negative keys saved in the dictionary from the last iteration. Straightforwardly, the keys at the current step from the momentum encoder can be used to replace the above negative keys. The results in the second row of Table 2 show that this replacement increases the performance. This (slight) performance boost can be attributed to the improved positive-negative consistency (note that the positive keys are from the momentum encoder at the current iteration). Through this replacement, we show that the dictionary in MoCo can be removed without performance drop; actually, the performance is improved by a visible margin. This new dictionary-free MoCo framework is termed SimMoCo, where “Sim” stands for “Simplified” and indicates the removal of the dictionary.

Dual temperature and its rationale. A key property of the SimMoCo is that it uses dual temperature for en-

abling independent control of intra-anchor and inter-anchor hardness-aware properties with τ_α and τ_β , respectively. As shown in Table 3, adopting a sufficiently large τ_β reduces the inter-anchor hardness-aware property and boosts the performance. On the other hand, the vector component does not allow such a sufficiently large temperature. This creates a dilemma choice of setting an appropriate single temperature. The rationale discussed here is supported by the results in Table 3. This rationale also somehow aligns with an observation in [57] that removing the positive pair from the denominator in the InfoNCE loss for decoupling the influence of positive sample and negative ones on each other. [57] justifies their decoupling from the perspective of learning efficiency, while our dual temperature highlights independent control of two hardness-aware properties.

Method	τ_α	τ_β	Accuracy (%)
ST	0.1	0.1	49.52
ST	1	1	32.09
DT	0.1	1	54.11
DT (reverse)	1	0.1	31.28

Table 3. Performance on CIFAR100 with different configurations of τ_α and τ_β . ST indicates *single temperature* with $\tau_\beta = \tau_\alpha$. DT indicates *dual temperature* with $\tau_\beta \neq \tau_\alpha$. DT (reverse) indicates a larger temperature is set for the vector component, *i.e.* $\tau_\alpha > \tau_\beta$.

5.2. Momentum Removal

SimCo. The momentum has been introduced in MoCo to increase the consistency of the dictionary [22]. Our proposed SimMoCo already has no dictionary, thus it might make sense to further remove the momentum from the SimMoCo for simplicity. Another merit of removing such momentum is that it allows the gradient to backward through the side of keys, which is empirically found to boost performance. The simplified momentum-free variant of SimMoCo is straightforwardly termed SimCo. In the SimCo, q and k are from the same encoder and they are symmetric with different augmentations. Assuming that the MBS is N , q and k both have N elements, for which q_i and k_j are positive samples to each other when $i = j$; otherwise they are negative samples to each other. The loss in Eq 7 does not allow gradient on k . To enable the gradient update on both q and k , we propose an alternative implementation for dual temperature. A new variant of InfoNCE with dual

temperature (DT) can be designed as:

$$\left\{ \begin{array}{l} \mathcal{L}_{q_i}^{DT} = -\text{sg}\left(\frac{W_\beta^i}{W_\alpha^i}\right) \log \frac{\exp(q_i \cdot k_i / \tau_\alpha)}{\sum_{j=1}^N \exp(q_i \cdot k_j / \tau_\alpha)} \\ W_\beta^i = 1 - \frac{\exp(q_i \cdot k_i / \tau_\beta)}{\sum_{j=1}^N \exp(q_i \cdot k_j / \tau_\beta)}, W_\alpha^i = 1 - \frac{\exp(q_i \cdot k_i / \tau_\alpha)}{\sum_{j=1}^N \exp(q_i \cdot k_j / \tau_\alpha)}, \end{array} \right. \quad (9)$$

where $\text{sg}(\frac{W_\beta^i}{W_\alpha^i})$ changes the temperature from τ_α to τ_β for the scalar component, while keeping τ_α in the vector component unchanged. Taking the symmetry into account, the final loss would be $(\mathcal{L}_{q_i} + \mathcal{L}_{k_i})/2$, where \mathcal{L}_{k_i} has the same form as \mathcal{L}_{q_i} but switches the position of q and k . With gradient update on both q and k , the loss in Eq 9 resembles that in [6] but uses half negative samples. More discussion on their relationship as well as the pseudo code for Eq 9 are in the supplementary.

6. Experimental Setup and Results.

6.1. Experimental Setup.

Training. Following the settings on *CIFAR experiment* in the official GitHub repository¹, we use SGD optimizer with momentum 0.9 and weight decay 5e-4, and the temperature is set to 0.1. We train each model for 200 epochs with the MBS of 256 on a single GPU. In the first 10 epochs, we use a linear warmup learning rate then decay learning rate following cosine decay schedule without restarts [32]. The highest learning rate is set to 0.03. The momentum coefficient is set to 0.99. The projector of baseline MoCo v2 consists of two linear layers with a ReLU activation function between them, for which we keep the same setting. The augmentations adopted are random color jittering, random horizontal flip, and random grayscale conversion. We highlight that for a fair comparison, *MoCo v2*, *SimMoCo*, and *SimCo* are always trained under the same setup except for the specified changes, such as the intended dual temperature and removal of dictionary and momentum. For the dual temperature, we need to set the temperature to different values. τ_α needs to be set to an appropriate value due to the so-called uniformity-tolerance dilemma [47]. We follow common setups to set τ_α to 0.1. For τ_β , we set it to 1.0. Empirically, we find that τ_β has no significant influence on the performance as long as it is set to a sufficiently large value for mitigating the inter-anchor hardness-aware property.

Evaluation. As shown in the solo-learn [11] frameworks, the performance gap between online and offline linear evaluation is not significant. For the convenience to avoid the need of retraining a linear classifier after the en-

¹<https://github.com/facebookresearch/moco>

coder pretraining, we directly report top-1 accuracy (%) on the validation dataset with the online linear evaluation.

6.2. Experimental Results

Temperature τ_α . Temperature has been identified as an important hyperparameter for controlling the balance between uniformity and tolerance [47]. With ResNet 18 on CIFAR100, the results with a wide range of τ_α are reported in Table 4. For all the three frameworks, a very small or a very large τ_α leads to inferior performance. Relatively, however, MoCo v2 is more sensitive to temperature variation, for which a detailed discussion is provided in the supplementary.

τ_α	0.05	0.1	0.5	1
MoCo V2	49.16	53.28	35.99	21.74
SimMoCo	53.67	54.11	42.25	32.42
SimCo	56.95	58.35	48.98	39.49

Table 4. Performance under different temperature settings of τ_α on the scalar component.

Batch size	64	128	256	512	1024
MoCo v2	52.58	54.40	53.28	51.47	48.90
SimMoCo	54.02	54.93	54.11	52.45	49.70
SimCo	58.04	58.29	58.35	57.08	55.34

Table 5. Performance comparison with different mini-batch sizes on CIFAR-100.

Mini-batch size. Here, we further investigate another important hyperparameter, mini-batch size (MBS) with the linear-scaling rule [18] adopted to change the learning rate proportional to the MBS. As shown in Table 5, we can observe that the proposed SimMoCo and SimCo achieve superior performance over a wide range of MBS. Notably, a smaller MBS leads to inferior performance for all the three frameworks, while a larger MBS does not always lead to a better performance, which can be attributed to training difficulty with a large MBS [58].

Longer training. We experiment with longer training and the results are shown in Table 6. The superiority of our simplified frameworks over the MoCo v2 can also be observed for longer epochs.

Epoch	200	400	800
MoCo v2	53.28	59.7	63.74
SimMoCo	54.11	60.11	63.82
SimCo	58.35	62.36	65.68

Table 6. Performance comparison for longer training.

Various architectures. On CIFAR100, we further compare the three frameworks with different architectures, including ResNet18, ResNet50, ViT tiny [14], Swin tiny [31]. The results in Table 7 suggest that SimMoCo consistently outperforms MoCo v2. SimCo consistently outperforms MoCo v2 as well as our SimMoCo by a large margin.

Architecture	ResNet-50	ViT tiny	Swin tiny	ResNet-18
MoCo v2	53.44	16.78	32	53.28
SimMoCo	54.64	21.35	33.07	54.11
SimCo	58.48	28.81	42.64	58.35

Table 7. Performance comparison on CIFAR-100 with different architectures.

Various dataset. With ResNet18, we evaluate on multiple datasets, including CIFAR-10, CIFAR-100, SVHN, STL10 and ImageNet-100. The results in Table 8 show that our simplified models consistently outperform the baseline MoCo v2, except that the performance of SimMoCo is slightly worse than that of MoCo v2. Despite the simplicity, SimCo generally performs the best among the three frameworks on all investigated datasets except for SVHN dataset, for which SimMoCo performs the best.

Dataset	CIFAR10	CIFAR100	SVHN	STL10	ImageNet100
MoCo v2	82.35	53.28	47.25	81.25	57.52
SimMoCo	82.36	54.11	53.67	80.56	58.2
SimCo	85.61	58.35	52.37	83.19	61.28
MoCo v2+	85.3	57.19	44.51	82.93	60.52
SimMoCo+	85.61	58.15	44.74	82.61	61.12

Table 8. Results w/ or w/o symmetric loss on various datasets. Note that by default SimCo adopts a symmetric loss.

Symmetric MoCo and SimMoCo. [9] has shown a symmetric loss leads to a performance boost for the frameworks with MoEn. Following [9], we term them MoCo v2+ and SimMoCo v2+ when the symmetric loss is adopted. The results in Table 8 show that the performance is boosted by a large margin. It is worth highlighting that unlike SimCo with a default symmetric loss, the symmetric loss in the MoEn-based MoCo v2+ and SimMoCo+ doubles the computation resources. Nonetheless, SimCo still outperforms them by a visible margin.

7. A Unified Perspective on SSL and Beyond

Currently, the CL frameworks can be roughly divided into two categories: (a) dictionary-free CL represented by SimCLR [6] and (b) dictionary-based CL represented by MoCo family. As shown in Figure 4, our proposed SimMoCo simplifies MoCo via dictionary removal and is further simplified into SimCo via momentum removal. SimCo

and SimCLR are both dictionary-free and momentum-free, and SimCLR can be roughly perceived as a special case of SimCo with the dual temperature set to the same value. In the following, we discuss how our investigation further brings a unified perspective on CL and non-CL frameworks.

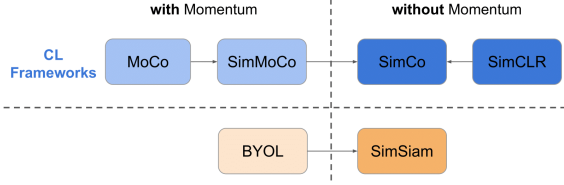


Figure 4. Comparison between CL and non-CL frameworks.

Type	Methods	Inter-anchor HA	Top-1 (%)
CL	SimMoCo	✓	50.22
	SimMoCo	✗	54.11
	SimCo	✓	53.82
	SimCo	✗	58.35
Non-CL	BYOL	✓	46.54
	BYOL	✗	50.65
	SimSiam	✓	39.18
	SimSiam	✗	51.78

Table 9. Influence of Inter-anchor hardness-awareness (HA) in CL and non-CL frameworks. All methods are based on ResNet-18, trained on CIFAR-100 with 200 epochs under the same setup.

7.1. Bridging the Gap Between CL and Non-CL

The SSL frameworks can be divided into CL and non-CL based on whether negative samples are used. For simplicity, we first discuss non-CL frameworks that use a simple loss:

$$\mathcal{L}_{q_i}^{ncl} = h(q_i) \cdot \text{sg}[-k_i], \quad (10)$$

where h is a prediction head. The above loss is adopted in BYOL [19] and SimSiam [8], which are arguably the two most popular non-CL frameworks. In practice, they use a symmetric loss and here for notation simplicity, we only take the non-symmetric loss into account. Comparing the loss in Eq 7 and that in Eq 10, we note a difference: the vanilla contrastive InfoNCE puts different penalty weights on anchors, while the non-CL frameworks [8, 19] adopt a loss that treats all anchors equally.

With the gradient on k disabled and τ_β set to sufficiently large, Eq 9 can be reformulated in the form of Eq 7 as:

$$\mathcal{L}_{q_i}^{cl} = q_i \cdot \text{sg}[-k_i + \sum_{j=1}^K \hat{p}_j^i k_j]. \quad (11)$$

where $\frac{1}{\tau_\alpha}$ is omitted for simple discussion as aforementioned. It is interesting to note that the above loss resembles Eq 2 for treating all anchors equally and resembles Eq 1 for keeping the intra-anchor hardness-awareness. Recently, [62] has shown that the negative samples in CL frameworks and predictor h in non-CL frameworks, SimSiam [8] for instance, achieve equivalent roles of de-centering and de-correlation for avoiding collapse. Some recent non-CL frameworks [4, 60] replace predictor with explicit de-correlation and regularization. In other words, their finding mainly bridges the gap between CL and non-CL frameworks from the perspective of intra-anchor hardness-awareness. Our work fills the gap by pointing out that the loss in non-CL frameworks [4, 8, 15, 19, 60] treats anchors equally, while vanilla contrastive InfoNCE in CL penalizes each anchor based on their hardness. Overall, through alleviating this imbalance (see the weight on k_i in Eq 11), our work further bridges the gap between CL and non-CL frameworks to have a unified understanding of SSL.

Inter-anchor hardness-awareness in non-CL. It is interesting whether inter-anchor hardness-awareness also affects non-CL frameworks. As aforementioned, non-CL frameworks treat anchors equally and thus we modify the loss in Eq 10 via multiplying it by $\text{sg}[\sum_{j=1}^K \hat{p}_j^i]$ for introducing inter-anchor hardness-awareness into non-CL. The results in Table 9 shows that such hardness-awareness also hurts the performance of both CL and non-CL frameworks.

7.2. Discussion

Inter-anchor hardness-awareness in SSL vs. SL. With the softmax function, cross-entropy (CE) loss in supervised learning (SL) also has the inter-anchor hardness-aware property. Our investigation suggests that unlike InfoNCE in SSL, such property is critical for competitive performance in SL. We find that this can be partly attributed to the explanation that this default anchor-wise weight is less reliable to indicate the hardness than that in SL. A more detailed discussion is provided in the supplementary.

8. Conclusion

In this work, we revisit MoCo family by analyzing its key component, namely momentum-based dictionary. Our extensive analysis reveals that such a large dictionary is required mainly due to an inter-anchor hardness-awareness property of the commonly used InfoNCE in CL. We propose to control two hardness-aware properties independently with dual temperature, which facilitates simplifying MoCo v2 through removing the dictionary as well as momentum. Extensive experiments have confirmed that our simplified frameworks, SimMoCo and SimCo, achieve competitive performance against their baseline MoCo V2. This work also bridges the gap between CL and Non-CL frameworks to form a unified understanding of SSL.

Acknowledgement

This work was partly supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) under grant No.2019-0-01396 (Development of framework for analyzing, detecting, mitigating of bias in AI model and training data), No.2021-0-01381 (Development of Causal AI through Video Understanding and Reinforcement Learning, and Its Applications to Real Environments) and No.2021-0-02068 (Artificial Intelligence Innovation Hub).

References

- [1] Aviad Aberdam, Ron Litman, Shahar Tsiper, Oron Anshel, Ron Slossberg, Shai Mazor, R. Manmatha, and Pietro Perona. Sequence-to-sequence contrastive learning for text recognition. In *CVPR*, 2021. 2
- [2] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019. 1, 2, 3
- [3] Yingbin Bai and Tongliang Liu. Me-momentum: Extracting hard confident examples from noisily labeled data. In *ICCV*, 2021. 1, 2
- [4] Adrien Bardes, Jean Ponce, and Yann LeCun. Vircreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021. 2, 8
- [5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021. 2
- [6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 1, 2, 6, 7
- [7] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2
- [8] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021. 2, 8
- [9] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *ICCV*, 2021. 2, 7
- [10] Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debaised contrastive learning. In *NeurIPS*, 2020. 2
- [11] Victor G. Turrissi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. Solo-learn: A library of self-supervised methods for visual representation learning, 2021. 6
- [12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019. 1
- [13] Ali Diba, Vivek Sharma, Reza Safdari, Dariush Lotfi, Saquib Sarfraz, Rainer Stiefelhagen, and Luc Van Gool. Vi2clr: Video and image for visual contrastive learning of representation. In *ICCV*, 2021. 2
- [14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2, 7
- [15] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In *ICML*. PMLR, 2021. 2, 8
- [16] Hyunjun Eun, Jinyoung Moon, Jongyoul Park, Chanho Jung, and Changick Kim. Learning to discriminate information for online action detection. In *CVPR*, 2020. 2
- [17] Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. In *ICLR*, 2020. 2
- [18] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large mini-batch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 7
- [19] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 2020. 2, 8
- [20] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010. 3
- [21] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 2
- [22] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 1, 2, 3, 4, 5, 6
- [23] Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *ICML*, 2020. 1, 2, 3
- [24] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017. 2
- [25] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 1, 2, 3
- [26] Chih-Hui Ho and Nuno Vasconcelos. Contrastive learning with adversarial examples. *arXiv preprint arXiv:2010.12050*, 2020. 1, 2
- [27] Hanzhe Hu, Jinshi Cui, and Liwei Wang. Region-aware contrastive learning for semantic segmentation. In *ICCV*, 2021. 2
- [28] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Mining on manifolds: Metric learning without labels. In *CVPR*, 2018. 1, 2

- [29] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *NeurIPS*, 2020. 2
- [30] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*, 2020. 1
- [31] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 7
- [32] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 6
- [33] Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah Erfani, and James Bailey. Normalized loss functions for deep learning with noisy labels. In *ICML*, 2020. 3
- [34] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *ECCV*, 2016. 2
- [35] Ping Nie, Yuyu Zhang, Xiubo Geng, Arun Ramamurthy, Le Song, and Daxin Jiang. Dc-bert: Decoupling question and document for efficient contextual encoding. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020. 1
- [36] Kento Nozawa and Issei Sato. Understanding negative samples in instance discriminative self-supervised representation learning. *arXiv preprint arXiv:2102.06866*, 2021. 1, 2
- [37] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 1, 2, 3
- [38] Tian Pan, Yibing Song, Tianyu Yang, Wenhao Jiang, and Wei Liu. Videomoco: Contrastive video representation learning with temporally adversarial examples. In *CVPR*, 2021. 2
- [39] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019. 1
- [40] Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *ICLR*, 2021. 2
- [41] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2
- [42] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NeurIPS*, 2016. 2, 3
- [43] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. {VL}-{bert}: Pre-training of generic visual-linguistic representations. In *ICLR*, 2020. 1
- [44] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 2, 3
- [45] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *ECCV 2020*, 2020. 1
- [46] Bhavya Vasudeva, Puneesh Deora, Saumik Bhattacharya, Ummapada Pal, and Sukalpa Chanda. Loop: Looking for optimal hard negative embeddings for deep metric learning. In *ICCV*, 2021. 1, 2
- [47] Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *CVPR*, 2021. 1, 2, 3, 5, 6, 7
- [48] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, 2020. 1
- [49] Weilun Wang, Wengang Zhou, Jianmin Bao, Dong Chen, and Houqiang Li. Instance-wise hard negative example generation for contrastive learning in unpaired image-to-image translation. In *ICCV*, 2021. 2
- [50] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. 2
- [51] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2021. 1
- [52] Haiyan Wu, Yanyun Qu, Shaohui Lin, Jian Zhou, Ruizhi Qiao, Zhizhong Zhang, Yuan Xie, and Lizhuang Ma. Contrastive learning for compact single image dehazing. In *CVPR*, 2021. 2
- [53] Mike Wu, Milan Mosse, Chengxu Zhuang, Daniel Yamins, and Noah Goodman. Conditional negative sampling for contrastive learning of visual representations. *arXiv preprint arXiv:2010.02037*, 2020. 2
- [54] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018. 1, 2, 3
- [55] Enze Xie, Jian Ding, Wenhao Wang, Xiaohang Zhan, Hang Xu, Peize Sun, Zhenguo Li, and Ping Luo. Detco: Unsupervised contrastive learning for object detection. In *ICCV*, 2021. 2
- [56] Yazhou Yao, Zeren Sun, Chuanyi Zhang, Fumin Shen, Qi Wu, Jian Zhang, and Zhenmin Tang. Jo-src: A contrastive approach for combating noisy labels. In *CVPR*, 2021. 2
- [57] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. *arXiv preprint arXiv:2110.06848*, 2021. 6
- [58] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017. 1, 7
- [59] Xumin Yu, Yongming Rao, Wenliang Zhao, Jiwen Lu, and Jie Zhou. Group-aware contrastive regression for action quality assessment. In *ICCV*, 2021. 2
- [60] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *ICML*, 2021. 2, 8
- [61] Kaiwei Zeng, Munan Ning, Yaohua Wang, and Yang Guo. Hierarchical clustering with hard-batch triplet loss for person re-identification. In *CVPR*, 2020. 1, 2
- [62] Chaoning Zhang, Kang Zhang, Chenshuang Zhang, Trung X Pham, Chang D Yoo, and In So Kweon. How does simsiam avoid collapse without negative samples? a unified understanding with self-supervised contrastive learning. In *ICLR*, 2022. 2, 3, 8
- [63] Oliver Zhang, Mike Wu, Jasmine Bayrooti, and Noah Goodman. Temperature as uncertainty in contrastive learning. *arXiv preprint arXiv:2110.04403*, 2021. 2, 3

- [64] Rui Zhu, Bingchen Zhao, Jingen Liu, Zhenglong Sun, and Chang Wen Chen. Improving contrastive learning by visualizing feature transformation. In *ICCV*, 2021. [2](#), [3](#)
- [65] Chengxu Zhuang, Tianwei She, Alex Andonian, Max Sobol Mark, and Daniel Yamins. Unsupervised learning from video with deep neural embeddings. In *CVPR*, 2020. [2](#)
- [66] Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *ICCV*, 2019. [1](#), [2](#), [3](#)

Supplementary Material

A. Setup of Figure 2 in the main manuscript

We train the model on CIFAR100 with MoCo v2 for 200 epochs on a single GPU. we use SGD optimizer with momentum 0.9 and weight decay $5e-4$, and the temperature is set to 0.1. We use a linear warmup learning rate then decay learning rate following cosine decay schedule without restarts. Here, we adopt two independent dictionaries, D_{vector} and D_{scalar} to store negative sample keys for vector and scalar components, respectively. We fix one of them to have the dictionary size of 65536, while changing the dictionary size of the other one. We also report the results of a single dictionary with various dictionary size in Fig. 5. As expected, the performance decreases significantly when the dictionary size is small.

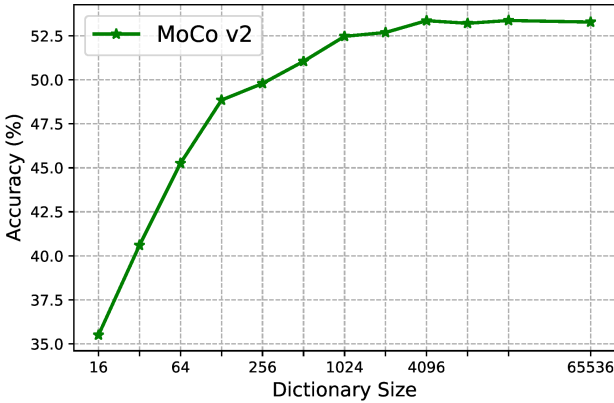


Figure 5. Influence of dictionary size in MoCo v2.

B. The scalar component is less sensitive to the quality of the keys

We also report the results for sampling a certain number (K_{scalar}) of keys from D_{scalar} while using the full D_{vector} . We set K_{scalar} to 4096, since our results in Figure 2 of the main manuscript show that the scalar component requires a sufficiently large dictionary for competitive performance. The results in Table 10 show that there is only a small performance gap among the three sampling strategies. Notably, the model still converges well with a reasonable performance even when the earliest keys are sampled, while the model does not converge for K_{vector} in the same setup. The results show that the scalar component is less sensitive to the key quality.

Sampling strategies	Earliest	Random	Newest
Top-1 Accuracy(%)	52.13	52.75	53.32

Table 10. Comparison of various sampling strategies on CIFAR100.

C. The pseudo code for the relationship of dual temperature

The core difference between the InfoNCE with dual temperature in Eq 9 of the main manuscript and that in [6] lies in whether dual temperature is applied. Moreover, the loss in [6] uses negative samples from both encoders, while InfoNCE with dual temperature uses only half negative sample. For example, when q_i is the anchor, it only uses negative samples from the encoder k side, which simplifies the code implementation. The pseudo code is shown in Algorithm 1. Adopting negative samples from both sides is confirmed to yield equivalent performance.

D. MoCo v2 is more sensitive to temperature variation

Note that MoCo v2 by default adopts a single temperature, *i.e.* $\tau_\beta = \tau_\alpha$. When the temperature is very small, the inter-anchor hardness-aware sensitivity gets higher, leading to lower performance, while our SimMoCo and SimCo have no such concerns because τ_β is large. When the temperature is very large, the dependence of MoCo v2 on the old keys gets higher, *i.e.* lower PN consistency. The PN consistency for our SimMoCo and SimCo is always optimal because the negative keys are generated by the same encoder as the positive keys. Thus, our SimMoCo and SimCo have no such consistency concerns as MoCo v2. Overall, we observe that our proposed SimMoCo and SimCo consistently outperform the baseline MoCo v2.

E. InfoNCE in SSL vs. CE in SL.

The CE loss in supervised learning (SL) is shown as

$$\mathcal{L}_{CE} = -\log \frac{\exp(\mathbf{o}_{gt}/\tau)}{\sum_{c=1}^C \exp(\mathbf{o}_c/\tau)}, \quad (12)$$

where \mathbf{o} indicates the network output which is a logit vector of length C (total number of classes) and gt indicates the index for the ground-truth (GT) class. Note that the sum is over the GT class and $(C-1)$ non-GT classes. With one hot vector defined as \mathbf{y} , there exists the following equivalence: $\mathbf{o}_{gt} = \mathbf{o} \cdot \mathbf{y}_{gt}$ and $\mathbf{o}_c = \mathbf{o} \cdot \mathbf{y}_c$.

Algorithm 1 Pytorch-like Pseudocode: Dual Temperature Loss

```
def simco_loss(query, key, intra_temperature,
               inter_temperature):
    """
    N: batch size
    D: the dimension of representation vector

    Args:
        query (torch.Tensor): NxD Tensor containing
            projected features from view 1.
        key (torch.Tensor): NxD Tensor containing
            projected features from view 2.
        intra_temperature (float): temperature factor
            for the intra component.
        inter_temperature (float): temperature factor
            for the inter component.

    Returns:
        torch.Tensor: SimCo loss.
    """
    # normalize query and key
    query = F.normalize(query, dim=-1)
    key = F.normalize(key, dim=-1)

    # calculate logits
    logits = query @ key.T

    # intra awareness
    logits_intra = logits / intra_temperature
    prob_intra = F.softmax(logits_intra, dim=1)

    # inter awareness
    logits_inter = logits / inter_temperature
    prob_inter = F.softmax(logits_inter, dim=1)

    # inter awareness changing factor
    mask = torch.ones(prob_inter.size()).
        fill_diagonal_(0)
    weight_alpha = (prob_intra * mask).sum(-1)
    weight_beta = (prob_inter * mask).sum(-1)

    inter_intra = weight_beta / weight_alpha

    # loss calculation
    log_softmax = F.log_softmax(logits, dim=-1)
    log_softmax_diag = log_softmax.diag()

    loss = -inter_intra.detach() * log_softmax_diag
    return loss.mean()
```

Based on the above equivalence, compared with Eq 1 in the main manuscript, we show that CE loss is a special case of InfoNCE by perceiving the GT one-hot vector as the positive key and other non-GT one-hot vectors as negative keys. With such a high resemblance between the two losses, however, unlike InfoNCE in SSL, this inter-anchor hardness-aware property is widely known to be important for competitive performance. In other words, alleviating the inter-anchor hardness-aware property does not help CE loss to improve the performance.

Here, we attempt to provide an intuitive explanation. Imagine that we do not have prior knowledge on the hardness of anchor sample, straightforwardly, the loss should be designed to treat every anchor sample equally. Given such prior knowledge, it is intuitive that the loss should put more weight on the hard anchor samples, such as CE does. Regarding this prior, the main difference between InfoNCE

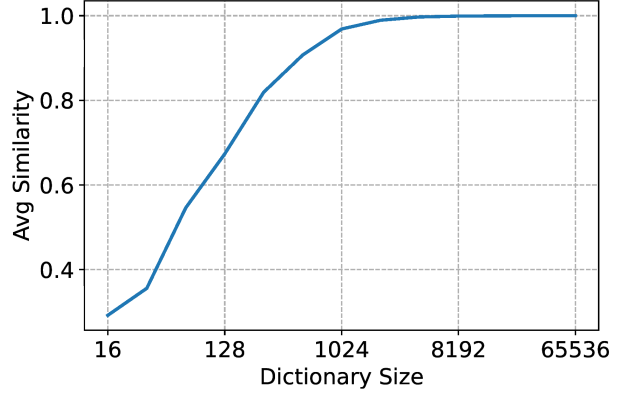


Figure 6. Cosine similarity between different r_+^i through changing the positive and negative keys randomly. Low similarity indicates that the inter-anchor hardness-aware weight is not reliable because a reliable prior should not deviate too much through changing the positive and negative keys.

and CE is that the prior knowledge in CE is very reliable because the keys (both GT and non-GT) are fixed yet correct. However, this prior is less reliable in the InfoNCE loss because the keys are random. For example, the positive key with the same image of another random augmentation, and the negative keys are encoded from the random images. By changing the positive and negative keys randomly, we get two sets of r_+^i (see Eq 8 in the main manuscript) and calculate their similarity. The results in Figure 6 show that the similarity is low when the dictionary size is small, indicating this inter-anchor weight is not reliable. Intuitively, if this prior is unreliable, this inter-anchor hardness-aware property is misleading and thus it might be better to decrease this hardness-aware property, *i.e.* treating every anchor sample equally as in our investigation.

Method	Symmetric			Asymmetric		
	0.4	0.6	0.8	0.4	0.6	0.8
CE	57.59	39.36	20.39	57.89	38.62	19.29
CE (DT)	63.95	56.21	22.51	63.07	59.53	21.8

Table 11. Test accuracy (%) of standard CE and CE (DT) on CIFAR10 with symmetric label noise ($\eta \in \{0.4, 0.6, 0.8\}$) and asymmetric label noise ($\eta \in \{0.4, 0.6, 0.8\}$).

With the above interpretation, the inter-anchor hardness-aware weight might also be detrimental to CE loss if the prior gets less reliable. A straightforward way to make the prior less reliable is to corrupt the data with noisy labels. We follow the setup in prior works [33] that study noisy labels. Specifically, the noise can be corrupted in a symmetric or asymmetric manner. The results with different noise ratios are shown in Table 11. We observe that CE with dual tem-

perature to remove the inter-anchor hardness-aware property outperforms the standard CE loss by a visible margin. Note that this experiment is conducted to prove our interpretation instead of pushing the SOTA performance in the setup of noisy labels.