EqCo: Equivalent Rules for Self-supervised Contrastive Learning

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we propose a method, named EqCo (Equivalent Rules for Contrastive Learning), to make self-supervised learning irrelevant to the number of negative samples in the contrastive learning framework. Inspired by the InfoMax principle, we point that the margin term in contrastive loss needs to be adaptively scaled according to the number of negative pairs in order to keep steady mutual information bound and gradient magnitude. EqCo bridges the performance gap among a wide range of negative sample sizes, so that we can use only a few negative pairs (e.g. 16 per query) to perform self-supervised contrastive training on large-scale vision datasets like ImageNet, while with almost no accuracy drop. This is quite a contrast to the widely used large batch training or memory bank mechanism in current practices. Equipped with EqCo, our simplified MoCo (SiMo) achieves comparable accuracy with MoCo v2 on ImageNet (linear evaluation protocol) while only involves 16 negative pairs per query instead of 65536, suggesting that large quantities of negative samples might not be a critical factor in contrastive learning frameworks.

1 Introduction and Background

Self-supervised learning has recently received much attention in the field of visual representation learning [Hadsell et al. (2006); Dosovitskiy et al. (2014); Oord et al. (2018); Bachman et al. (2019); Henaff et al. (2019); Wu et al. (2018); Tian et al. (2019); He et al. (2020); Misra & Maaten (2020); Grill et al. (2020); Cao et al. (2020); Tian et al. (2020)], as its potential to learn universal representations from unlabeled data. Among various self-supervised methods, one of the most promising research paths is contrastive learning [Oord et al. (2018)], which has been demonstrated to achieve comparable or even better performances than supervised training for many downstream tasks such as image classification, object detection, and semantic segmentation [Chen et al. (2020); He et al. (2020); Chen et al. (2020a,b)].

The core idea of contrastive learning is briefly summarized as follows: first, extracting a pair of embedding vectors \((q(I), k(I))\) (named query and key respectively) from the two augmented views of each instance \(I\); then, learning to maximize the similarity of each positive pair \((q(I), k(I))\) while pushing the negative pairs \((q(I), k(I'))\) (i.e., query and key extracted from different instances accordingly) away from each other. To learn the representation, an InfoNCE loss [Oord et al. (2018); Wu et al. (2018)] is conventionally employed in the following formulation (slightly modified with an additional margin term):

\[
\mathcal{L}_{NCE} = \frac{1}{q \sim \mathcal{D}, k_0 \sim \mathcal{D}', k_i \sim \mathcal{D}'} \left[ -\log \frac{e^{q^\top k_0 - m}}{e^{q^\top k_0 - m} + \sum_{i=1}^{K} e^{q^\top k_i}} \right] ,
\]

where \(q\) and \(k_i (i = 0, \ldots, K)\) stand for the query and keys sampled from the two (augmented) data distributions \(\mathcal{D}\) and \(\mathcal{D}'\) respectively. Specifically, \(k_0\) is associated to the same instance as \(q\)’s while other \(k_i\)’s not; hence we name \(k_0\) and \(k_i (i > 0)\) positive sample and negative samples respectively in the remaining text, in which \(K\) is the number of negative samples (or pairs) for each query. The temperature \(\tau\) and the margin \(m\) are hyper-parameters. In most previous works, \(m\) is trivially set to zero (e.g. Oord et al. (2018); He et al. (2020); Chen et al. (2020a); Tian et al. (2020)) or some
handcraft values (e.g., Xie et al. (2020)). In the following text, we mainly study contrastive learning frameworks with InfoNCE loss as in Eq. 1 unless otherwise specified.  

In contrastive learning research, it has been widely believed that enlarging the number of negative samples $K$ boosts the performance (Henaff et al. (2019); Tian et al. (2019); Bachman et al. (2019)). For example, in MoCo (He et al., 2020) the ImageNet accuracy rises from 54.7% to 60.6% under linear classification protocol when $K$ grows from 256 to 65536. Such observation further drives a line of studies how to effectively optimize under a number of negative pairs, such as memory bank methods (Wu et al., 2018; He et al., 2020) and large batch training (Chen et al., 2020a), either of which empirically reports superior performances when $K$ becomes large. Analogously, in the field of supervised metric learning (Deng et al., 2019; Wang et al., 2018; Sun et al., 2020; Wang et al., 2020), loss in the similar form as Eq. 1 is often applied on a lot of negative pairs for hard negative mining. Besides, there are also a few theoretical studies supporting the viewpoint. For instance, Oord et al. (2018) points out that the mutual information between the positive pair tends to increase with the number of negative pairs $K$; Wang & Isola (2020) find that the negative pairs encourage features’ uniformity on the hypersphere; Chuang et al. (2020) suggests that large $K$ leads to more precise estimation of the debiased contrastive loss; etc.

Despite the above empirical or theoretical evidence, however, we point out that the reason for using many negative pairs is still less convincing. First, unlike the metric learning mentioned above, in self-supervised learning, the negative terms $k_i$ in Eq. 1 include both “true negative” (whose underlying class label is different from the query’s, similarly hereinafter) and “false negative” samples, since the actual ground truth label is not available. So, intuitively large $K$ should not always be beneficial because the risk of false negative samples also increases (known as class collision problem). Arora et al. (2019) thus theoretically concludes that a large number of negative samples could not necessarily help. Second, some recent works have proven that by introducing new architectures (e.g., a predictor network in BYOL (Grill et al., 2020), or designing new loss functions (e.g., Caron et al. (2020a); Tarnov et al. (2020)), state-of-the-art performance can still be obtained even without any explicit negative pairs. In conclusion, it is still an open question whether large quantities of negative samples are essential to contrastive learning.

After referring to the above two aspects, we rise a question: is a large $K$ really essential in the contrastive learning framework? We propose to rethink the question from a different view: note that in Eq. 1 there are three hyper-parameters: the number of negative samples $K$, temperature $\tau$, and margin $m$. In most of previous empirical studies (He et al., 2020; Chen et al., 2020a), only $K$ is changed while $\tau$ and $m$ are usually kept constant. Do the optimal hyper-parameters of $\tau$ and $m$ varies with $K$? If so, the performance gains observed from larger $K$s may be a wrong interpretation – merely brought by suboptimal hyper-parameters’ choices for small $K$s, rather than much of an essential.

In the paper, we investigate the relationship among three hyper-parameters and suggest an equivalent rule:

$$m = \tau \log \frac{\alpha}{K},$$

where $\alpha$ is a constant. We find that if the margin $m$ is adaptively adjusted based on the above rule, the performance of contrastive learning is irrelevant to the size of $K$, in a very large range (e.g. $K \geq 16$). For example, in MoCo framework, by introducing EqCo the performance gap between $K = 256$ and $K = 65536$ (the best configuration reported in He et al. (2020)) almost disappears (from 6.1% decrease to 0.2%). We call this method “Equivalent Rules for Contrastive learning” (EqCo). For completeness, as the other part of EqCo we point that adjusting the learning rate according to the conventional linear scaling rule satisfies the equivalence for different number of queries per batch.

Theoretically, following the InfoMax principle (Linsker (1988)) and the derivation in CPC (Oord et al., 2018), we prove that in EqCo, the lower bound of the mutual information keeps steady under various numbers of negative samples $K$. Moreover, from the back-propagation perspective, we further prove that in such configuration the upper bound of the gradient norm is also free of

---

1Recently, some self-supervised learning algorithms achieve new state-of-the-art results using different frameworks instead of conventional InfoNCE loss as in Eq. 1 (e.g., mean teacher in BYOL (Grill et al., 2020) and online clustering (in SWAV Caron et al., 2020b). We will investigate them in the future.
The proposed equivalent rule implies that, by assigning $\alpha = K_0$, it can “mimic” the optimization behavior under $K_0$ negative samples even if the physical number of negatives $K \neq K_0$.

The “equivalent” methodology of EqCo follows the well-known linear scaling rule (Krizhevsky, 2014; Goyal et al., 2017), which suggests scaling the learning rate proportional to the batch size if the loss satisfies with the linear averaged form: $L = \frac{1}{N} \sum_{i=1}^{N} f(x_i; \theta)$. However, linear scaling rule cannot be directly applied on InfoNCE loss (Eq. 1), which is partially because InfoNCE loss includes two batch sizes (number of queries and keys respectively) while linear scaling rule only involves one, in addition to the nonlinearity of the keys in InfoNCE loss. In the experiments of SimCLR (Chen et al., 2020a), learning rates under different batch sizes are adjusted with linear scaling rule, but the accuracy gap is still very large (57.5%@batch=256 vs. 64+%@batch=8192, 100 epochs training).

EqCo challenges the belief that self-supervised contrastive learning requires large quantities of negative pairs to obtain competitive performance, making it possible to design simpler algorithms. We thus present SiMo, a simplified contrastive learning framework based on MoCo v2 (Chen et al., 2020c). SiMo is elegant, efficient, free of large batch training and memory bank; moreover, it can achieve superior performances over state-of-the-art even if the number of negative pairs is extremely small (e.g. 16), without bells and whistles.

The contributions of our paper are summarized as follows:

- We challenge the widely accepted belief that on large-scale vision datasets like ImageNet, large size of negative samples is critical for contrastive learning. We interpret it from a different view: it may be because the hyper-parameters are not set to the optimum.
- We propose EqCo, an equivalent rule to adaptively set hyper-parameters between small and large numbers of negative samples, which proves to bridge the performance gap.
- We present SiMo, a simpler but stronger baseline for contrastive learning.

2 EqCo: Equivalent Rules for Contrastive Learning

In this section we introduce EqCo. We mainly consider the circumstance of optimizing the InfoNCE loss (Eq. 1) with SGD. For each batch of training, there are two meanings of the concept “batch size”, i.e., the size of negative samples/pairs $K$ per query, and the number of queries (or positive pairs) $N$ per batch. Hence our equivalent rules accordingly consist of two parts, which will be introduced in the next subsections.

2.1 The Case of Negative Pairs

Our derivation is mainly inspired by the model of Contrastive Predictive Coding (CPC) (Oord et al., 2018), in which InfoNCE loss is interpreted as a mutual information estimator. We further extend the method so that it is applicable to InfoNCE loss with a margin term (Eq. 1), which is not considered in Oord et al. (2018).

Following the concept in Oord et al. (2018), given a query embedding $q$ (namely the context in Oord et al. (2018)) and suppose $K + 1$ random key embeddings $x = \{x_i\}_{i=0,\ldots,K}$, where there exists exactly one entry (e.g., $x_i$) sampled from the conditional distribution $P(x_i|q)$ while others (e.g., $x_j$) sampled from the “proposal” distribution $P(x_j)$ independently. According to which entry corresponds to the conditional distribution, we therefore defines $K + 1$ candidate distributions for $x$ (denoted by $\{H_i\}_{i=0,\ldots,K}$), where the probability density of $x$ under $H_i$ is $P_{H_i}(x) = P(x_i|q) \prod_{j \neq i} P(x_j)$. So, given the observed data $X = \{x_0,\ldots,x_K\}$ of $x$, the probability where $x$ is sampled from $H_0$ rather than other candidates is thus derived with Bayes theorem:

$$
\text{Pr}[x \sim H_0|q, X] = \frac{P^+P_{H_0}(X)}{P^+P_{H_0}(X) + P^- \sum_{i=1}^{K} P_{H_i}(X)} = \frac{P^+ P_{H_0}(X)}{P^+ P_{H_0}(X) + \sum_{i=1}^{K} P_{H_i}(X)}
$$

(2)
where we denote $P^+$ and $P^-$ as the prior probabilities of $H_0$ and $H_i$ ($i > 0$) respectively. We point that Eq. 2 introduces a generalized form to that in Oord et al. [2018] by taking the priors into account. Referring to the notations in Eq. 1 we suppose that $H_0$ is the ground truth distribution of $x$ (since $k_0$ is the only positive sample). By modeling the density ratio $P(k_i|q)/P(k_i)$ as $e^{q^T k_i/\tau} (i = 0, \ldots, K)$ and letting $P^+/P^- = e^{-m/\tau}$, the negative log-likelihood $\mathcal{L}_{opt} \triangleq \mathbb{E}_{q,X} \log \Pr[x \sim H_0|q,X]$ can be regarded as the optimal value of $\mathcal{L}_{NCE}$.

Similar to the methodology of Oord et al. [2018], we explore the lower bound of $\mathcal{L}_{opt}$:

$$
\mathcal{L}_{opt} = \mathbb{E}_{q \sim D, k_0 \sim D'(q), k_i \sim D'} \log \left( 1 + e^{m/\tau} \frac{P(k_0)}{P(k_0|q)} \sum_{i=1}^{K} \frac{P(k_i|q)}{P(k_i)} \right) 
\approx \mathbb{E}_{q \sim D, k_0 \sim D'(q)} \log \left( 1 + \frac{Ke^{m/\tau}}{P(k_0)} \frac{P(k_0|q)}{P(k_0)} \sum_{i=1}^{K} \frac{P(k_i|q)}{P(k_i)} \right) 
= \mathbb{E}_{q \sim D, k_0 \sim D'(q)} \log \left( 1 + Ke^{m/\tau} \frac{P(k_0|q)}{P(k_0)} \right) 
\geq \log(1 + Ke^{m/\tau}) - \mathcal{I}(k_0, q),
$$

where $\mathcal{I}(\cdot, \cdot)$ means mutual information. The approximation in the second row is guaranteed by Law of Large Numbers as well as the fact $P(k_i|q) \approx P(k_i)$ since $k_i$ ($i > 0$) and $q$ are “almost” independent. The inequality in the last row is resulted from $P(k_0|q) \geq P(k_0)$ as $k_0$ and $q$ are extracted from the same instance. Therefore the lower bound of the mutual information (noted as $f_{bound}(m, K)$) between the positive pair $(k_0, q)$ is:

$$
\mathcal{I}(k_0, q) \geq f_{bound}(m, K) \triangleq \log(1 + Ke^{m/\tau}) - \mathcal{L}_{opt} 
\approx \log(1 + Ke^{m/\tau}) - \mathbb{E}_{q \sim D, k_0 \sim D'(q)} \log \left( 1 + Ke^{m/\tau} \frac{P(k_0)}{P(k_0|q)} \right). \quad (4)
$$

So, minimizing $\mathcal{L}_{NCE}$ (Eq. 1) towards $\mathcal{L}_{opt}$ implies maximizing the lower bound of the mutual information, which is also satisfied when $m \neq 0$. In the case of $m = 0$, the result is consistent with that in Oord et al. [2018]. Oord et al. [2018] further points out the bound increases with $K$, which indicates larger $K$ encourages to learn more mutual information thus could help to improve the performance.

Nevertheless, different from Oord et al. [2018] our model does not require $m$ to be zero, so the lower bound in Eq. 3 is also a function of $e^{m/\tau}$. Thus we have the following theorem:

**Theorem 1. (Main, EqCo for negative pairs)** The mutual information lower bound of InfoNCE loss in Eq. 1 is irrelevant to the number of negative pairs $K$, if

$$
m = \tau \log \frac{\alpha}{K}, \quad (5)
$$

where $\alpha$ is a constant coefficient. And in the circumstances the bound is given by:

$$
f_{bound} \left( \tau \log \frac{\alpha}{K}, K \right) \approx \log(1 + \alpha) - \mathbb{E}_{q \sim D, k_0 \sim D'(q)} \log \left( 1 + \alpha \frac{P(k_0)}{P(k_0|q)} \right) \approx f_{bound}(0, \alpha), \quad (6)
$$

which can be immediately obtained by substituting Eq. 5 into Eq. 4. We name Eq. 5 as “equivalent condition”.

Theorem 1 suggests a property of equivalency: under the condition of Eq. 5 no matter what the number of physical negative pairs $K$ is, the optimal solution of $\mathcal{L}_{NCE}$ (Eq. 1) is “equivalent” in the sense of the same mutual information lower bound. The bound is controlled by a hyper-parameter $\alpha$ rather than $K$. Eq. 6 further implies that the lower bound also correlates to the configuration of $K = \alpha$ without margin, which suggests we can “mimic” the InfoNCE loss’s behavior of $K = K_0$ under a different physical negative sample size $K_1$, just by applying Eq. 5 with $\alpha = K_0$. It inspires us to simplify the existing state-of-the-art frameworks (e.g. MoCo [He et al. (2020)]) with fewer negative samples but as accurate as the original configurations, which will be introduced next.

We empirically validate Theorem 1 as follows. Notice that $f_{bound}$ is difficult to calculate directly because $\mathcal{L}_{opt}$ is not known. Instead, we plot the empirical mutual information lower bound
Given the equivalent condition (Eq. \ref{eq:equivalent}) and a query embedding \( \mathbf{q} \) as well as the corresponding positive sample \( \mathbf{k}_0 \), for \( \mathcal{L}_{NCE} \) in Eq. \ref{eq:nce_loss} the expectation of the gradient norm w.r.t. \( \mathbf{q} \) is bounded by \[^{2}\]\[ 
\mathbb{E}_{\mathbf{k}_i \sim \mathcal{D}^+} \left\| \frac{d\mathcal{L}_{NCE}}{d\mathbf{q}} \right\| 
\leq \frac{2}{\tau} \left( 1 - \frac{\exp(\mathbf{q}^\top \mathbf{k}_0/\tau)}{\exp(\mathbf{q}^\top \mathbf{k}_0/\tau) + \alpha \mathbb{E}_{\mathbf{k}_i \sim \mathcal{D}^+} \left[ \exp(\mathbf{q}^\top \mathbf{k}_i/\tau) \right]} \right). 
\]  
\[^{2}\]Some works (e.g., He et al. (2020)) only use \( d\mathcal{L}_{NCE}/d\mathbf{q} \) for optimization. In contrast, other works (Chen et al. (2020a)) also involve \( d\mathcal{L}_{NCE}/dk_i \) for optimization. In contrast, other works (Chen et al. (2020a)) also involve \( d\mathcal{L}_{NCE}/dk_i \) for optimization. In contrast, other works (Chen et al. (2020a)) also involve \( d\mathcal{L}_{NCE}/dk_i \) for optimization. In contrast, other works (Chen et al. (2020a)) also involve \( d\mathcal{L}_{NCE}/dk_i \) for optimization. In contrast, other works (Chen et al. (2020a)) also involve \( d\mathcal{L}_{NCE}/dk_i \) for optimization.
Theorem 2 indicates that, equipped with the equivalent rule (Eq. 5), the upper bound of the gradient norm is irrelevant to the number of negative samples $K$. Fig. 2 (see the Appendix A.2) further validates our theory: the gradient norm becomes much more steady after using EqCo under different $K$s. Since the size of $K$ affects little on the gradient magnitude, gradient scaling techniques, e.g. linear scaling rule, are not required specifically for different $K$s. Eq. 7 also implies that the temperature $\tau$ significantly affects the gradient norm even EqCo is applied – it is why we only recommend to modify $m$ for equivalence (Eq. 5), though the mutual information lower bound is determined by $e^{m/\tau}$ as a whole.

2.2 The Case of Positive Pairs

In practice the InfoNCE loss (Eq. 1) is usually optimized with batched SGD, which can be represented as empirical risk minimization:

$$\mathcal{L}_{NCE}^{\text{batch}} = \frac{1}{N} \sum_{j=1}^{N} \mathcal{L}_{NCE}^{(j)}(q_j, k_{j,0}),$$

(8)

where $N$ is the number of queries (or positive pairs) per batch; $(q_j, k_{j,0}) \sim (\mathcal{D}, \mathcal{D}'(q_j))$ is the $j$-th positive pair, and $\mathcal{L}_{NCE}^{(j)}(q_j, k_{j,0})$ is the corresponding loss. For different $j$, $\mathcal{L}_{NCE}^{(j)}$ is (almost) independent of each other, because $q_j$ is sampled independently. Hence, Eq. 8 satisfies the form of linear scaling rule (Krizhevsky (2014); Goyal et al. (2017)), suggesting that the learning rate should be adjusted proportional to the number of queries $N$ per batch.

Remarks 3. Previous work like SimCLR (Chen et al. (2020a)) also proposes to apply linear scaling rule. The difference is, in SimCLR it does not clarify the concept of “batch size” refers to the number of queries or the number of keys. However in our paper, we explicitly point that the linear scaling rule needs to be applied corresponding to the number of queries per batch ($N$) rather than $K$.

2.3 Empirical Evaluation

In this subsection we conduct experiments on the three state-of-the-art self-supervised contrastive learning frameworks – MoCo (He et al. (2020)), MoCo v2 (Chen et al. (2020c)) and SimCLR (Chen et al. (2020a)) to verify our theory in Sec. 2.1 and Sec. 2.2. We propose to alter $K$ and $N$ separately to examine the correctness of our equivalent rules.

Implementation details. We follow most of the training and evaluation settings recommended in the original papers respectively. The only difference is, for SimCLR, we adopt SGD with momentum rather than LARS (You et al. (2017)) as the optimizer. We use ResNet-50 (He et al. (2016)) as the default network architecture. 128-d features are employed for query and key embeddings. Unless specially mentioned, all models are trained on ImageNet (Deng et al. (2009)) for 200 epochs without using the ground truth labels. We report the top-1 accuracy under the conventional linear evaluation protocol according to the original paper respectively. The number of queries per batch ($N$) is set to 256 by default. All models are trained with 8 GPUs.

It is worth noting the way we alter the number of negative samples $K$ independent of $N$ during training. For MoCo and MoCo v2, we simply need to set the size of the memory bank to $K$. Specially, if $K < N$, in the current batch the memory bank is actually composed of $K$ random keys sampled from the previous batch. While for SimCLR, if $K < N$ we random sample $K$ negative keys for

---

1In SimCLR, the authors find that square-root learning rate scaling is more desirable with LARS optimizer (You et al. (2017)), rather than linear scaling rule. Also, their experiments suggest that the performance gap between large and small batch sizes become smaller under that configuration. We point that the direction is orthogonal to our equivalent rule. Besides, SimCLR does not explore the case of very small $K$'s (e.g. $K <= 128$).
Under review as a conference paper at ICLR 2021

Figure 2: Comparisons with/without EqCo under different number of negative samples (noted by $K$). Results are evaluated with ImageNet top-1 accuracy using linear evaluation protocol. In EqCo, we set $\alpha = 65536$ for MoCo and MoCo v2, and $\alpha = 256$ for SimCLR (except for one data point with $\alpha = 4096$, as noted in the legend). Best viewed in color.

Quantitative results. Fig. 2 illustrates the effect of our equivalent rule under different $K$s. Our experiments start with the best configurations (i.e. $K = 65536$ for MoCo and MoCo v2, and $K = 256$ for SimCLR), then we gradually reduce $K$ and benchmark the performance. Results in Fig. 2 indicates that, without EqCo the accuracy significantly drops if $K$ becomes very small (e.g. $K < 64$). While with EqCo, by setting $\alpha$ to “mimic” the optimal $K$, the performance surprisingly keeps steady under a wide range of $K$s. Fig. 2(b) further shows that in SimCLR, by setting $\alpha$ to a number larger than the physical batch size (e.g. 4096 vs. 256), the accuracy significantly improves from 62.0% to 65.3%, suggesting the benefit of EqCo especially when the memory is limited. The comparison fully demonstrates EqCo is essential especially when the number of negative pairs is small.

Besides, Table 1 compares the results of MoCo v2 under different number of queries $N$, while $K = 65536$ is fixed. It is clear that, with linear scaling rule (Krizhevsky (2014); Goyal et al. (2017)), the final performance is almost unchanged under different $N$, suggesting the effectiveness of our equivalent rule for $N$.

<table>
<thead>
<tr>
<th>$N$ ($K = 65536$)</th>
<th>256</th>
<th>512</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top-1 accuracy (%)</td>
<td>67.5</td>
<td>67.5</td>
<td>67.4</td>
</tr>
</tbody>
</table>

Table 1: ImageNet accuracy (MoCo v2) vs. the number of queries per batch ($N$). The learning rates during training are adjusted with linear scaling rule.

3 SiMo: A Simpler but Stronger Baseline

EqCo inspires us to rethink the design of contrastive learning frameworks. The previous state-of-the-arts like MoCo and SimCLR heavily rely on large quantities of negative pairs to obtain high accuracy. In the original paper of SimCLR (Chen et al. (2020a)), the best number of negative pairs is around 4096. However, the largest $K$ we can use in our experiment is 256 due to GPU memory limit.

Our “mimicking” result (65.3%, $\alpha = 4096$, $K = 256$) is slightly lower than the counterpart score reported in the original SimCLR paper (66.6%, with a physical batch size of $K = 4096$), which we think may be resulted from the extra benefits of SyncBN along with LARS optimizer used in SimCLR, especially when the physical batch size is large.
performances, hence implementation tricks such as memory bank and large batch training are introduced, which makes the system complex and tends to be costly. Thanks to EqCo, we are able to design a simpler contrastive learning framework with fewer negative pairs.

<table>
<thead>
<tr>
<th>Method</th>
<th>Epochs</th>
<th>Top-1 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPC v2 (Henaff et al., 2019)</td>
<td>200</td>
<td>63.8</td>
</tr>
<tr>
<td>CMC (Tian et al., 2019)</td>
<td>240</td>
<td>66.2</td>
</tr>
<tr>
<td>SimCLR (Chen et al., 2020a)</td>
<td>200</td>
<td>66.6</td>
</tr>
<tr>
<td>MoCo v2 (Chen et al., 2020c)</td>
<td>200</td>
<td>67.5</td>
</tr>
<tr>
<td>InfoMin Aug. (Tian et al., 2020)</td>
<td>200</td>
<td>70.1</td>
</tr>
<tr>
<td>SiMo ($K = 16, \alpha = 256$)</td>
<td>200</td>
<td>68.1</td>
</tr>
<tr>
<td>SiMo ($K = 256, \alpha = 256$)</td>
<td>200</td>
<td>68.0</td>
</tr>
<tr>
<td>SiMo ($K = 256, \alpha = 65536$)</td>
<td>200</td>
<td>68.5</td>
</tr>
<tr>
<td>PIRL (Misra &amp; Maaten, 2020)</td>
<td>800</td>
<td>63.6</td>
</tr>
<tr>
<td>SimCLR (Chen et al., 2020a)</td>
<td>1000</td>
<td>69.3</td>
</tr>
<tr>
<td>MoCo v2 (Chen et al., 2020c)</td>
<td>800</td>
<td>71.1</td>
</tr>
<tr>
<td>InfoMin Aug. (Tian et al., 2020)</td>
<td>800</td>
<td>73.0</td>
</tr>
<tr>
<td>SiMo ($K = 256, \alpha = 256$)</td>
<td>800</td>
<td>71.8</td>
</tr>
<tr>
<td>SiMo ($K = 256, \alpha = 65536$)</td>
<td>800</td>
<td>72.1</td>
</tr>
</tbody>
</table>

Table 2: State-of-the-art InfoNCE-based frameworks

We propose SiMo, a simplified variant of MoCo v2 (Chen et al., 2020c) equipped with EqCo. We follow most of the design in Chen et al. (2020c), where the key differences are as follows:

**Memory bank.** MoCo, MoCo v2 and SimCLR v2 (Chen et al. (2020b)) employ memory bank to maintain large number of negative embeddings $k_i$, in which there is a side effect: every positive embedding $k_0$ is always extracted from a “newer” network than the negatives’ in the same batch, which could harm the performance. In SiMo, we thus cancel the memory bank as we only rely on a few negative samples per batch. Instead, we use the momentum encoder to extract both positive and negative key embeddings from the current batch.

**Shuffling BN vs. Sync BN.** In MoCo v1/v2, shuffling BN (He et al. (2020)) is proposed to remove the obvious dissimilarities of the BN (Ioffe & Szegedy (2015)) statistics between the positive (from current mini-batch) and the negatives (from memory bank), so that the model can make predictions based on the semantic information of images rather than the BN statistics. In contrast, since the positive and negatives are from the same batch in SiMo, therefore, we use sync BN (Peng et al. (2018)) for simplicity and more stable statistics. Sync BN is also used in SimCLR (Chen et al. (2020a)) and SimCLR v2 (Chen et al. (2020b)).

There are a few other differences, including 1) we use a BN attached to each of the fully-connected layers; 2) we introduce a warm-up stage at the beginning of the training, which follows the methodology in SimCLR (Chen et al., 2020a). Apart from all the differences mentioned above, the architecture and the training (including data augmentations) details in SiMo are exactly the same as MoCo v2’s. In the following text, the number of queries per batch (N) is set to 256, and the backbone network is ResNet-50 by default.

**Quantitative results.** First, we empirically demonstrate the necessity of EqCo in SiMo framework. We choose the number of negative samples $K = 256$ as the baseline, then reduce $K$ to evaluate the performance. Fig. 3 shows the result on ImageNet using linear evaluation protocol. Without EqCo, the accuracy significantly drops when $K$ is very small. In contrast, using EqCo to “mimic” the case of large $K$ (by setting $\alpha$ to 256), the accuracy almost keeps steady even under very small $K$s.

Table 2 further compares our SiMo with state-of-the-art self-supervised contrastive learning methods on ImageNet. Using only 16 negative samples per query, SiMo outperforms MoCo v2 (68.1% vs. 67.5%). If we increase $\alpha$ to 65536 to “simulate” the case under huge number of negative pairs, the accuracy further increases to 68.5%. Moreover, when we extend the training epochs to 800, we get the accuracy of 72.1%, surpassing the baseline MoCo v2 by 1.0%. The only entry that surpasses...
our results is *InfoMin Aug.* (Tian et al. (2020)), which is mainly focused on data generation and orthogonal to ours. The experiments indicate that SiMo is a simpler but more powerful baseline for self-supervised contrastive learning. Readers can refer to the Appendix B for more experimental results of SiMo.

4 LIMITATIONS AND FUTURE WORK

Theorem 1 suggests that given the equivalent condition (Eq. 5), InfoNCE losses under various $K$s are “equivalent” in the sense of the same mutual information lower bound, which is also backed up with the experiments in Fig. 1. However, Fig. 2 (a) shows that if $K$ is smaller than a certain value (e.g. $K \leq 16$), some frameworks like MoCo v2 start to degrade significantly even with EqCo; while for other frameworks like SiMo (Fig. 3), the accuracy almost keeps steady for very small $K$s. Tschannen et al. (2019) also point that the principle of InfoMax cannot explain all the phenomena in contrastive learning. We will investigate the problem in the future, e.g. from other viewpoints such as gradient noise brought by small $K$s (Fig. 4 in Appendix A.2 gives some insights).

Though the formulation of Eq. 1 is very common in the field of supervised metric learning, which is usually named margin softmax cross-entropy loss (Deng et al., 2019; Wang et al., 2018; Sun et al., 2020). Nevertheless, unfortunately, our equivalent rule seems invalid to be generalized to those problems (e.g. face recognition). The major issue lies in the approximation in Eq. 3, we need the negative samples $k_i$ to be independent of the query $q$, which is not satisfied in supervised tasks.

According to Fig. 2 and Fig. 3, the benefits of EqCo become significant if $K$ is sufficiently small (e.g. $K < 64$). But in practice, for modern computing devices (e.g. GPUs) it is not that difficult to use $\sim 256$ negative pairs per query. Applying EqCo to “simulate” more negative pairs via adjusting $\alpha$ can further boost the performance, however, whose accuracy gains become relatively marginal. For example, in Table 2 under 200 epochs training, SiMo with $\alpha = 65536$ outperforms that of $\alpha = 256$ by only 0.5%. It could be a fundamental limitation of InfoNCE loss. We will investigate the problem in the future.
REFERENCES


A DETAILS ABOUT THEOREM 2

A.1 PROOF OF EQ. [7]

Given the equivalent condition (Eq. 5) and a query embedding \( q \) as well as the corresponding positive sample \( k_0 \), for \( \mathcal{L}_{\text{NCE}} \) in Eq. 1 the expectation of the gradient norm w.r.t. \( q \) is bounded by:

\[
\mathbb{E}_{k_i \sim \mathcal{D}'} \left\| \frac{\text{d}\mathcal{L}_{\text{NCE}}}{\text{d}q} \right\| \leq \frac{2}{\tau} \left( 1 - \frac{\exp(q^\top k_0/\tau)}{\exp(q^\top k_0/\tau) + \alpha \mathbb{E}_{k_i \sim \mathcal{D}'}[\exp(q^\top k_i/\tau)\big]} \right), \tag{9}
\]

**Proof.** For simplicity, we denote the term \( \exp(q^\top k_i/\tau) \) as \( s_i (i = 0, \ldots, K) \). Then \( \mathcal{L}_{\text{NCE}} \) can be rewritten as:

\[
\mathcal{L}_{\text{NCE}} = -\log s_0 = \frac{s_0}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \tag{10}
\]

The gradient of \( \mathcal{L}_{\text{NCE}} \) with respect to \( q \) is easily derived:

\[
\frac{\text{d}\mathcal{L}_{\text{NCE}}}{\text{d}q} = -\frac{1}{\tau} \left( 1 - \frac{s_0}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \right) k_0 + \frac{\alpha}{\tau K} \sum_{i=1}^{K} \frac{s_i}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} k_i, \tag{11}
\]

Owing to the *Triangle Inequality* and the fact that \( k_i (i = 0, \ldots, K) \) is normalized, the norm of gradient is bounded by:

\[
\left\| \frac{\text{d}\mathcal{L}_{\text{NCE}}}{\text{d}q} \right\| \leq \frac{1}{\tau} \left( 1 - \frac{s_0}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \right) \cdot \|k_0\| + \frac{\alpha}{\tau K} \sum_{i=1}^{K} \frac{s_i}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \cdot \|k_i\| = \frac{1}{\tau} \left( 1 - \frac{s_0}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \right) + \frac{\alpha}{\tau K} \sum_{i=1}^{K} \frac{s_i}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \tag{12}
\]

Since the cosine similarity between \( q \) and \( k_i (i = 1, \ldots, K) \) is bounded in \([-1, 1]\), we know the expectation of \( \mathbb{E}_{k_i \sim \mathcal{D}'}[s_i] \) exists. According to Inequality (12) and *Jensen’s Inequality*, we have:

\[
\mathbb{E}_{k_i \sim \mathcal{D}'} \left[ \frac{\frac{2}{\tau} \left( 1 - \frac{s_0}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \right)}{\frac{2}{\tau} \left( 1 - \frac{s_0}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \right)} \right] = \frac{2}{\tau} \left( 1 - \mathbb{E}_{k_i \sim \mathcal{D}'} \left[ \frac{s_0}{s_0 + \frac{\alpha}{K} \sum_{i=1}^{K} s_i} \right] \right) \tag{13}
\]

Replacing \( s_i \) by \( \exp(q^\top k_i/\tau) \), the proof of Theorem 2 is completed.

\[\square\]
Figure 4: The means (solid line) and variances (ribbon, \(\pm \sigma\)) of \(\|d_L^{NCE}/dq\|\) under different \(K\)s. We train a normal MoCo v2 for 200 epochs and show the statistics at different epochs.
B  MORE EXPERIMENTS ON SiMo

For the following experiments of this section, we report the top-1 accuracy of SiMo on ImageNet (Deng et al., 2009) under the linear evaluation protocol. The backbone of SiMo is ResNet-50 (He et al., 2016) and we train SiMo for 200 epochs unless noted otherwise.

B.1 ABLATION ON MOMENTUM UPDATE

In MoCo (He et al., 2020) and MoCo v2 (Chen et al., 2020c), the key encoder is updated by the following rule:

\[ \theta_k = \beta \theta_k + (1 - \beta) \theta_q \]

where \( \theta_q \) and \( \theta_k \) stand for the weights of query encoder and key encoder respectively, and \( \beta \) is the momentum coefficient. For SiMo, we also adopt the momentum update and use the key encoder to compute the features of positive sample and negative samples.

In Table 3, we report the results of SiMo with different momentum coefficients. The number of training epochs is set to be 100, so the top-1 accuracy of baseline (\( \beta = 0.999 \)) drops to 64.4%. Compared to the baseline, SiMo without momentum update (\( \beta = 0 \)) is inferior, showing the advantage of momentum update.

<table>
<thead>
<tr>
<th>( \beta )</th>
<th>0</th>
<th>0.999</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy (%)</td>
<td>62.1</td>
<td>64.4</td>
</tr>
</tbody>
</table>

Table 3: Ablation on momentum update.

B.2 ABLATION ON BN

Table 4 shows the performance of SiMo equipped with shuffling BN or Sync BN. Likewise, we train SiMo for 100 epochs. It is easy to check out that SiMo with shuffling BN struggles to perform well. Besides, compared to MoCo v2, SiMo with shuffling BN degrades significantly, and we conjecture that it is because the MLP structure of SiMo is more suitable for Sync BN, rather than shuffling BN.

<table>
<thead>
<tr>
<th></th>
<th>Shuffling BN</th>
<th>Sync BN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy (%)</td>
<td>58.8</td>
<td>64.4</td>
</tr>
</tbody>
</table>

Table 4: Sync BN vs. shuffling BN.

B.3 SIJo WITH DIFFERENT \( \alpha \)

As shown in Sec 2.1, \( \alpha \) is related to the lower bound of mutual information. Table 5 reveals how accuracy of SiMo varies with the choice of \( \alpha \). As we increase \( \alpha \) to 65536, the accuracy tends to improve, in accordance with the Eq 6. However, when \( \alpha \) is too large (e.g., 262144), the performance slightly drops by 0.2%.

<table>
<thead>
<tr>
<th>( \alpha )</th>
<th>256</th>
<th>1024</th>
<th>4096</th>
<th>16384</th>
<th>65536</th>
<th>262144</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy (%)</td>
<td>68.0</td>
<td>68.1</td>
<td>68.1</td>
<td>68.4</td>
<td>68.5</td>
<td>68.3</td>
</tr>
</tbody>
</table>

Table 5: SiMo with different \( \alpha \).

Similar results can be found in MoCo v2. We increase \( K \) to 262144 in MoCo v2, the accuracy also descends (in Table 6).
B.4 SiMo with Wider Models

Results using wider models are presented in Table 7. For SiMo, the performance is further boosted with wider models (more channels). For instance, SiMo with ResNet-50 (2x) and ResNet-50 (4x) outperforms the baseline (68.5%) by 2% and 3.8% respectively.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Param. (M)</th>
<th>α</th>
<th>Top-1 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-50 (2x)</td>
<td>94</td>
<td>256</td>
<td>70.2</td>
</tr>
<tr>
<td>ResNet-50 (2x)</td>
<td>94</td>
<td>65536</td>
<td>70.5</td>
</tr>
<tr>
<td>ResNet-50 (4x)</td>
<td>375</td>
<td>256</td>
<td>71.9</td>
</tr>
<tr>
<td>ResNet-50 (4x)</td>
<td>375</td>
<td>65536</td>
<td>72.3</td>
</tr>
</tbody>
</table>

Table 7: SiMo with wider models. All models are trained with 200 epochs.

B.5 Transfer to Object Detection

Setup

We utilize FPN [Lin et al., 2017] with a stack of $4 \times 3$ convolution layers in R-CNN head to validate the effectiveness of SiMo. Following the MoCo training protocol, we fine-tune with synchronized batch-normalization [Peng et al., 2018] across GPUs. The additional initialized layers are also equipped with BN for stable training. To effectively validate the transferability of the features, the training schedule is set to be 12 epochs (known as $1 \times$), in which learning rate is initialized as 0.2 and decreased at 7 and 11 epochs with a factor of 0.1. The image scales are random sampled of [640, 800] pixels during training and fixed with 800 at inference.

Results

Table 8 summarizes the fine-tuning results on COCO val2017 of different pre-training methods. Random initialization indicates training COCO from scratch, and supervised represents conventional pre-training with ImageNet labels. Compared with MoCo, SiMo achieves competitive performance without large quantities of negative pairs. It is also on a par with the supervised counterpart and significantly outperforms random initialized one.

<table>
<thead>
<tr>
<th>pre-train</th>
<th>AP</th>
<th>AP$_{50}$</th>
<th>AP$_{75}$</th>
<th>AP$_s$</th>
<th>AP$_m$</th>
<th>AP$_l$</th>
</tr>
</thead>
<tbody>
<tr>
<td>random init</td>
<td>31.4</td>
<td>49.4</td>
<td>34.0</td>
<td>17.9</td>
<td>32.3</td>
<td>41.6</td>
</tr>
<tr>
<td>supervised</td>
<td>39.0</td>
<td>59.1</td>
<td>42.6</td>
<td>22.4</td>
<td>42.2</td>
<td>50.6</td>
</tr>
<tr>
<td>MoCo v2</td>
<td>39.1</td>
<td>59.2</td>
<td>42.5</td>
<td>23.3</td>
<td>42.1</td>
<td>50.8</td>
</tr>
<tr>
<td>SiMo</td>
<td>39.0</td>
<td>59.2</td>
<td>42.3</td>
<td>22.9</td>
<td>41.8</td>
<td>50.5</td>
</tr>
</tbody>
</table>

Table 8: Object detection fine-tuned on COCO.

C A Toy Evaluation of EqCo

To evaluate the effectiveness of EqCo as mutual information (MI) estimator, following the configuration of Poole et al., 2019, we estimate the MI lower bound of between two simple random vectors.

Specifically, given that $(X, Y)$ are drawn from the known correlated Gaussian distribution, we calculate the lower bound of MI between $X$ and $Y$ based on their embedding. $X$ is a 20-dimensional random variables drawn from a standard Gaussian distribution. And we sampled $Y$ with the following rule:

$$Y = \rho X + \sqrt{1 - \rho^2} \epsilon$$  \hspace{1cm} (14)
where $\rho$ is the given correlation coefficient and $\epsilon$ is a random variable sampled from a standard Gaussian distribution and independent from $X$. With a known $\rho$, the ground truth MI between $X$ and $Y$ is easy to compute:

$$I(X, Y) = -\frac{d}{2} \log (1 - \rho^2)$$

(15)

Here, $d$ is the dimension of $X$ and $Y$, and as mentioned above we set $d = 20$.

To embed $X$ and $Y$, we adopt two MLPs respectively, and each MLP has 1 hidden layer of 256 units, followed by ReLU activation function. We use Adam optimizer with learning rate of 0.0005 to optimize InfoNCE or EqCo for 5000 steps. For each training iteration, $K$ pairs of $(X, Y)$ are independently sampled, which means there are $K-1$ negative samples for each query. After training, the weights of MLPs are frozen and we repeat estimating the lower bound of MI for 1000 times to reduce the estimating variance. For experiments with EqCo, we set the $\alpha = 512$.

As shown in Table 9, $I_{\text{NCE}}$ varies with $K$, while $I_{\text{EqCo}}$ remains steady. Especially, when the ground truth MI is relatively large (e.g., 8, 10), significant differences between EqCo and InfoNCE can be observed. The experiment further validates the effectiveness of EqCo.

<table>
<thead>
<tr>
<th>Mutual Information</th>
<th>$K$</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_{\text{NCE}}$</td>
<td>2.0</td>
<td>1.7</td>
<td>1.8</td>
<td>1.9</td>
<td>1.9</td>
</tr>
<tr>
<td>$I_{\text{EqCo}}$</td>
<td>1.9</td>
<td>1.9</td>
<td>1.9</td>
<td>1.9</td>
<td></td>
</tr>
<tr>
<td>$I_{\text{NCE}}$</td>
<td>4.0</td>
<td>2.9</td>
<td>3.2</td>
<td>3.4</td>
<td>3.6</td>
</tr>
<tr>
<td>$I_{\text{EqCo}}$</td>
<td>3.8</td>
<td>3.7</td>
<td>3.6</td>
<td>3.6</td>
<td></td>
</tr>
<tr>
<td>$I_{\text{NCE}}$</td>
<td>6.0</td>
<td>3.6</td>
<td>4.1</td>
<td>4.5</td>
<td>4.9</td>
</tr>
<tr>
<td>$I_{\text{EqCo}}$</td>
<td>5.1</td>
<td>5.0</td>
<td>4.9</td>
<td>4.9</td>
<td></td>
</tr>
<tr>
<td>$I_{\text{NCE}}$</td>
<td>8.0</td>
<td>3.9</td>
<td>4.6</td>
<td>5.1</td>
<td>5.6</td>
</tr>
<tr>
<td>$I_{\text{EqCo}}$</td>
<td>5.8</td>
<td>5.7</td>
<td>5.7</td>
<td>5.6</td>
<td></td>
</tr>
<tr>
<td>$I_{\text{NCE}}$</td>
<td>10.0</td>
<td>4.1</td>
<td>4.7</td>
<td>5.4</td>
<td>6.0</td>
</tr>
<tr>
<td>$I_{\text{EqCo}}$</td>
<td>6.1</td>
<td>6.0</td>
<td>6.0</td>
<td>6.0</td>
<td></td>
</tr>
</tbody>
</table>

Table 9: Estimating mutual information by InfoNCE and EqCo with different batch size and various ground truth mutual information.