STOCHASTIC APPROXIMATION TO CONTRASTIVE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Contrastive learning has proven to be a powerful paradigm for self-supervised representation learning, yet traditional methods often rely on arbitrary definitions of positive and negative pairs, requiring large batch sizes to manage the tradeoff between contrastive terms effectively. This approach wastes significant computational resources on negative pairs that contribute minimal learning signals. To address these limitations, we propose a novel method that reformulates contrastive learning as a matrix approximation problem using I-divergence, a non-normalized variant of Kullback-Leibler divergence. Our objective function is decomposable across instance pairs, enabling efficient stochastic approximation algorithms that perform well with fewer negative samples by leveraging neighbor embeddings. Additionally, we generalize the scaling factor beyond standard normalization to adaptively emphasize positive pairs with higher learning potential, reducing computational waste from negative pairs. Experimental results on benchmark datasets such as CIFAR and ImageNet demonstrate that our method outperforms existing contrastive learning approaches, particularly with small batch sizes and as few as one negative pair, highlighting its effectiveness and computational efficiency.

1 INTRODUCTION

Learning meaningful representations is fundamental to the success of machine learning systems (Bengio et al., 2013; Goodfellow et al., 2016). While conventional supervised approaches have dominated in recent years, driven by the increasing availability of labeled data, they face scalability challenges and are constrained by the need for extensive labeled examples (LeCun et al., 2015; Balestriero et al., 2023). Self-supervised learning (SSL) has emerged as a transformative paradigm, leveraging the inherent structure of unlabeled data to generate pseudo-labels (Liu et al., 2023; Jing & Tian, 2020; de Sa, 1993; Goyal et al., 2019). In domains such as computer vision, SSL has demonstrated the capability to rival or even surpass supervised pretraining in effectiveness (Tomasev et al., 2022; Goyal et al., 2019; He et al., 2020; Caron et al., 2020; Misra & Maaten, 2020).

Despite its promise, most contrastive learning methods, a cornerstone of SSL, require large batch sizes to manage the balance between positive and negative pairs, leading to significant computational inefficiencies. Specifically, substantial resources are spent on negative pairs that contribute minimal learning signals. Recent advancements, such as SogCLR (Yuan et al., 2022), address some of these inefficiencies by employing an exponentially moving average (EMA) within a effectively decoupled contrastive learning objective (DCL; Yeh et al., 2022), omitting the explicit positive-pair term. However, SogCLR introduces additional complexity by requiring EMA-updated scalars for each data instance, which can become inconvenient for large datasets.

To overcome these challenges and enable efficient mini-batch training, we reformulate contrastive learning as a matrix approximation problem using a non-normalized Kullback-Leibler divergence with a scaling factor. Our proposed objective function is decomposable across instance pairs, allowing the development of efficient stochastic approximation algorithms that perform effectively with significantly fewer negative samples. Furthermore, we establish a theoretical connection between our approach and SimCLR while extending beyond it by generalizing the scaling factor. This generalization enables dynamic prioritization of positive pairs that provide richer learning signals, thereby reducing computational waste associated with negative pairs and improving overall efficiency.

We conducted experiments on widely recognized vision benchmark datasets, including CIFAR and ImageNet, to assess the effectiveness of our approach. Our method was evaluated against several state-of-the-art contrastive learning techniques, demonstrating consistent superiority. The results show that our approach is not only more computationally efficient—requiring smaller training batches and fewer negative pairs—but also achieves higher accuracy and overall performance compared to competing methods. These findings underscore the potential of our approach as a cost-effective and high-performing solution for vision-related tasks.

2 Background and related work

2.1 Contrastive methods

Contrastive Learning (CL) (Bromley et al., 1993; Hadsell et al., 2006; Chopra et al., 2005; Gutmann & Hyvärinen, 2010; Mikolov et al., 2013; Oord et al., 2018; Chen et al., 2020a; Sohn, 2016) has historically been influential in representation learning and is based on push- and pull mechanisms between representations. There are supervised approaches (Khosla et al., 2020) which finds positive pairs based on class labels and unsupervised SSL approaches (Chen et al., 2020a; He et al., 2020; Caron et al., 2020; Wu et al., 2018; Misra & Maaten, 2020; Dosovitskiy et al., 2014) which generate positive pairs.

The Contrastive loss (Bromley et al., 1993; Hadsell et al., 2006; Chopra et al., 2005) and Triplet loss (Schroff et al., 2015; Weinberger & Saul, 2009) persisted on mining strategies for tricky negative pairs to be effective. The Triplet loss configured triplets of one positive and negative sample and was generalized to M negative samples instead in Sohn (2016). SimCLR (Chen et al., 2020a) applied this contrastive loss in SSL, which is similar to CPC's (Oord et al., 2018) InfoNCE, in a full batch mode setting where the other positive pairs from the batch configured negative pairs. However SimCLR's CL-loss is not decomposable in minibatch-optimization mode because the negative pairs are weighted relatively in the minibatch (Chen et al., 2022; 2020a). An unfortunate effect of this is that it becomes vulnerable in minibatch-optimization, and this degradation of performance is shown in Chen et al. (2020a). Several CL approaches have resorted to memory-banks (He et al., 2020; Chen et al., 2020b; Caron et al., 2020; Misra & Maaten, 2020) or unreasonable batchsizes (Chen et al., 2020a) and there are sampling strategies (Kalantidis et al., 2020; Robinson et al., 2020), but this can be computationally challenging.

Improving the affordability of contrastive representation learning is still an active research area, and recent related work include among others Yeh et al. (2022); Yuan et al. (2022); Qiu et al. (2023); Chen et al. (2022); Sharma et al. (2023); Shah et al. (2021); HaoChen et al. (2021). DeCL (Chen et al., 2022) consider the non-decomposibility of SimCLR's CL-loss and address the gradient issues. A decomposable spectral contrastive loss was proposed by HaoChen et al. (2021). DCL (Yeh et al., 2022) propose to remove the positive pair from the negative forces which show faster convergence and better results over CL baselines with/without memory-banks. SogCLR (Yuan et al., 2022) propose a global contrastive loss by mixing a zero-initialized running average with a local minibatch-estimate of the negative pairs and shows improvement over SimCLR on ImageNet and CLIP (Radford et al., 2021) in the multi-modal setting. SogCLR is improved in Qiu et al. (2023) with individualized temperatures. AUC-CL (Sharma et al., 2023) propose to combine contrastive learning with AUC maximization.

2.2 Non-contrastive methods

The contrastive methods rely on direct comparisons between positive and negative pairs (Jaiswal et al., 2020) to avoid a degenerate solution which is contrary to the non-contrastive methods which can leverage the positive pairs without directly contrasting to negative pairs (Garrido et al., 2023). Distillation methods (Grill et al., 2020; Chen & He, 2021; Caron et al., 2021; Zhou et al., 2022) exclusively use positive pairs and avoid degenerate solution by adaptions of the architecture while covariance based (Zbontar et al., 2021; Bardes et al., 2021; Ermolov et al., 2021) decorrelate embeddings over the embedding-space (Garrido et al., 2023). The masked learning approaches (He et al., 2022; Zhou et al., 2022) are not based on *multi-view invariance* but typically require certain encoder architectures. Despite self-distillation methods have been effective in SSL there is missing theoretical foundation, and in some areas the architecture cannot be used for example multi-modal

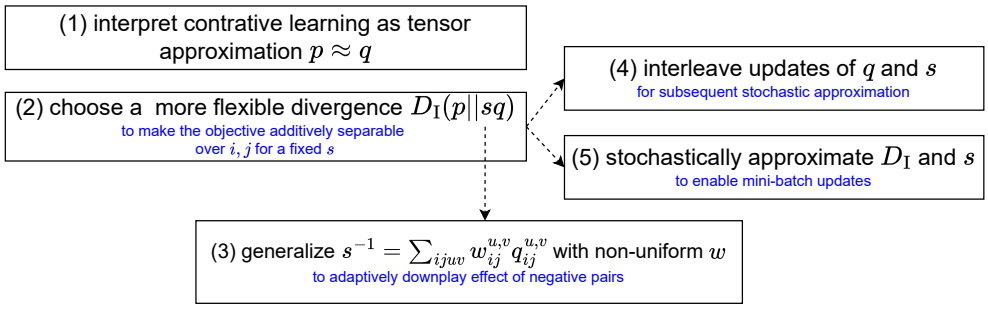


Figure 1: Development clue of our method.

representation learning (Radford et al., 2021; Qiu et al., 2023) and for recent domain agnostic approaches (Sui et al., 2024), where contrastive approaches are more effective.

2.3 Neighbor embedding methods

Neighbor embedding (Hinton & Roweis, 2002) (NE) is set of approaches which use the neighborhood graph from the input space as pseudo-labels for the mapped representations. Two pairwise probability matrices P and Q denotes the neighborhood graphs where P_{ij} and Q_{ij} respectively gives the probability of i and j being neighbor in the input-space and representation-space (Hinton & Roweis, 2002; van der Maaten & Hinton, 2008). To obtain this approximation a divergence $D(P||Q)$ is minimized between the pairwise probabilities. Initially SNE (Hinton & Roweis, 2002) normalized the probabilities row-wise which means minimizing a sum of divergences $D(P||Q) = \sum_i D(P_{i:}||Q_{i:})$. In t-SNE (van der Maaten & Hinton, 2008) the probabilities are symmetric, normalized matrix-wise and minimized over one divergence typically Kullback-Leilber divergence $D(P||Q) = \sum_{i \neq j} P_{ij} \log \frac{P_{ij}}{Q_{ij}}$.

It is conceivable a practical difficulty to compute the normalized probabilities P and Q. Most solutions can evaluate P offline however the probabilities in Q have to be re-normalized each iteration of optimization. Sampling based approaches UMAP (McInnes et al., 2018), TriMap (Amid & Warmuth, 2019), LargeVis (Tang et al., 2016), SCE (Yang et al., 2023) and PaCMAP (Wang et al., 2021) are not subject to this and can in many cases have better performance than t-SNE (McInnes et al., 2018; Wang et al., 2021; Damrich et al., 2023).

Contrastive learning aspects of NE-method where discussed at three papers at the ICLR 2023 conference by Böhm et al. (2023); Damrich et al. (2023); Hu et al. (2023). Damrich et al. (2023); Hu et al. (2023) discuss how SimCLR's contrastive loss in SSL can be thought of as a parametric t-SNE where the data augmentations are treated as sampling from a nearest graph and show that SimCLR with a Cauchy distribution often used in t-SNE can be very effective for NLDR on complex images where a euclidean metric is inadequate (Bengio et al., 2013). PCA and contrastive learing have been discussed in Tian (2022), and spectral methods and SSL have been brought up in (Balestriero & LeCun, 2022; HaoChen et al., 2021; Tan et al., 2024; Garrido et al., 2023).

3 Our method

3.1 Review of Stochastic Cluster Embedding

Our method is inspired by Stochastic Cluster Embedding (SCE; Yang et al., 2023), a similarity-based nonlinear dimensionality reduction (NLDR) method. SCE first computes a similarity matrix p that encodes the pairwise similarities between the high-dimensional data instances $\{\mathbf{x}_i\}_{i=1}^N$. Then it finds low-dimensional embedding $\{\mathbf{y}_i\}_{i=1}^N$ such that the pairwise similarities in the embedding space $q_{ij} = q(\mathbf{y}_i, \mathbf{y}_j)$ are close to those in the high-dimensional space. Usually $q_{ij} = q(\mathbf{y}_i, \mathbf{y}_j)$ is defined to be $\exp(-\|\mathbf{y}_i - \mathbf{y}_j\|^2)$ or $\frac{1}{1+\|\mathbf{y}_i - \mathbf{y}_j\|^2}$.

SCE employs a more flexible matrix divergence than the Kullback-Leibler (KL) in t-SNE:

$$D_{\mathrm{I}}(p||sq) = \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \left[p_{ij} \log \frac{p_{ij}}{sq_{ij}} - p_{ij} + sq_{ij} \right], \tag{1}$$

where $s > 0$ is a scaling factor. The divergence is called non-normalized KL-divergence or I-divergence. It measures the discrepancy between two matrices up to a scale. When $s^{-1} = \sum_{ij} q_{ij}$, minimizing the I-divergence reduces to minimizing the normalized KL-divergence. Differently, SCE periodically updates $s^{-1} = \sum_{ij, i\neq j} w_{ij} q_{ij}$, where $w_{ij} = \alpha p_{ij} N(N-1) + (1-\alpha)$ with $\alpha \in [0, 1]$. When $\alpha = 0$, it reduces to the t-SNE choice. When $\alpha > 0$, it mixes p and uniform sampling in calculating s, which adaptively reduces repulsion and thus improves cluster visualization (Yang et al., 2023).

3.2 Contrastive Learning Loss Function

SCE cannot be directly applied to contrastive learning due to several limitations: (1) SCE relies on fixed input similarities and cannot generalize to data points outside the training set; (2) its similarities are typically computed using simple metrics like Euclidean distances, which are inadequate for complex data objects such as images; and (3) SCE lacks mini-batch training algorithms tailored for contrastive learning. To address these issues, we introduce three key modifications: (1) we parameterize the embedding function using a deep neural network, $\mathbf{y} = f(\mathbf{x}; \theta)$, where θ represents the network weights; (2) we define similarities based on data and augmented view indices, eliminating the dependency on Euclidean or similar metrics for neighbor search before embedding; and (3) we propose stochastic approximation algorithms for the resulting learning objective and scaling factor. The progression of our development is illustrated in Figure 1.

For each input $\mathbf{x}_i$, we obtain its two augmented views $\tilde{\mathbf{x}}_i^{(1)}$ and $\tilde{\mathbf{x}}_i^{(2)}$ (one of the augmentation can be identity transform). Denote $\tilde{\mathbf{y}}_i^{(u)} = f\left(\tilde{\mathbf{x}}_i^{(u)}; \theta\right)$ and $q_{ij}^{u,v} = q\left(\tilde{\mathbf{y}}_i^{(u)}, \tilde{\mathbf{y}}_j^{(v)}\right)$ for $i, j \in \{1, \ldots, N\}$ and $u, v \in \{1, 2\}$. In contrastive learning, a well-trained neural network should make the outputs $\tilde{\mathbf{y}}_i^{(1)}$ and $\tilde{\mathbf{y}}_i^{(2)}$ from the same input instance $\mathbf{x}_i$ to be similar and the outputs from different instances (i.e., $i \neq j$) to be dissimilar. If we specify the similar target to be 1 and dissimilar to be 0, we can write the targets in a four-dimensional tensor $p \in \mathbb{R}^{N\times N\times 2\times 2}$, where $p_{ii}^{1,2} = p_{ii}^{2,1} = 1$ for $i = 1, \ldots, N$ and otherwise 0. Alternatively, we can reorgainze the tensors p and q in two $2N \times 2N$ matrices where $\psi_{2(i-1)+u, 2(j-1)+v} = p_{ij}^{u,v}$ and $\phi_{2(i-1)+u, 2(j-1)+v} = q_{ij}^{u,v}$. Then we can formulate contrastive learning as a matrix approximation problem by minimizing $D_{\mathrm{I}}(\psi||s\phi)$.

We neglect the matrix diagonal of ψ and ϕ in the approximation because $q_{ii}^{u,u}$'s are always a constant for Gaussian or Cauchy kernels. For notational simplicity, we set $p_{ii}^{u,u} = q_{ii}^{u,u} = 0$ for $i = 1, \ldots, N$ and $u = 1, 2$, i.e., $\psi_{aa} = \phi_{aa} = 0$ for $a = 1, \ldots, 2N$, which is equivalent to excluding them from the summations.

Because there are only a few nonzeros in ψ, we can rewrite $D_{\mathrm{I}}(\psi||s\phi)$ as

$$\mathcal{L}_{\mathrm{CLR}}(\theta) = \sum_{a=1}^{2N} \sum_{b=1}^{2N} \left[\psi_{ab} \log \frac{\psi_{ab}}{s\phi_{ab}} - \psi_{ab} + s\phi_{ab} \right] \tag{2}$$

$$= \sum_{i=1}^{N} -2 \log q_{ii}^{1,2} + s \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{u=1}^{2} \sum_{v=1}^{2} q_{ij}^{u,v} + C_1, \tag{3}$$

where $C_1 = -2N - 2N \log s$ is a constant for a fixed s. Following SCE, we periodically updates $s^{-1} = \sum_{ijuv} w_{ij}^{u,v} q_{ij}^{u,v}$ with

$$w_{ij}^{u,v} = \alpha p_{ij}^{u,v} N + (1-\alpha) = \begin{cases} \alpha N + (1-\alpha) & \text{when } i = j \\ 1 - \alpha & \text{otherwise.} \end{cases} \tag{4}$$

When $\alpha = 0$, s^{-1} becomes $\sum_{ijuv} q_{ij}^{u,v}$ and leads to normalized KL-divergence between ψ and ϕ.

This generalization brought by w can adaptively change the tradeoff between the first two terms in $\mathcal{L}_{\mathrm{CLR}}(\theta)$. At the training start, the non-zero q entries do not differ much, and thus $s^{-1} \approx \sum_{ijuv} q_{ij}^{u,v}$.

After minimizing the discrepancy for a while, $q_{ii}^{u,v}$'s ($u \neq v$) will become larger because of approximating p. When $\alpha > 0$, the matrix w concentrates more on the entries where $i = j$, and the resulting s^{-1} becomes larger than the uniform choice of w (i.e., $\alpha = 0$ or the normalized KL). Therefore, the generalization with $\alpha > 0$ dynamically emphasizes the first term, which contains the learning signals p, and downplays the second term without learning signals.

3.3 Minibatch-Mode Optimization

It is expensive to directly optimize the CL objective because it requires all pairs of data instances. Stochastic approximation is needed to facilitate minibatch-mode optimization. We first rewrite $\mathcal{L}_{\text{CLR}}(\theta)$ in an expectation manner

$$\mathcal{L}_{\text{CLR}}(\theta) = N\mathbb{E}_{i\sim\text{Uniform}(\{1,\dots,N\})}\left\{-2\log q_{ii}^{1,2} + sN\mathbb{E}_{j\sim\text{Uniform}(\{1,\dots,N\})}\left\{\sum_{u=1}^{2}\sum_{v=1}^{2} q_{ij}^{u,v}\right\}\right\}. \quad (5)$$

Our method, named Stochastic Approximation to Contrastive Learning (SACLR), minimizes the following minibatch-mode objective

$$\mathcal{L}_{\text{SACLR}}(\theta) = \sum_{i\in\mathcal{B}}\left[-2\log q_{ii}^{1,2} + s\frac{N}{M}\sum_{j\in\mathcal{M}_i}\sum_{u=1}^{2}\sum_{v=1}^{2} q_{ij}^{u,v}\right], \quad (6)$$

where $\mathcal{B} = \{i_1, \dots, i_B\} \subset [1, \dots, N]$, $\mathcal{M}_i \subseteq \mathcal{B}$, and $M = |\mathcal{M}_i|$ for all i. We have studied two choices of $\mathcal{M}_i$'s: (1) SACLR-1 where $M = 1$, and the objective simplifies to $\sum_{i\in\mathcal{B}}\left[-2\log q_{ii}^{1,2} + sN\sum_{uv} q_{ij}^{u,v}\right]$ with $j \sim \mathcal{B}$; and (2) SACLR-all where $M = B$ uses all negative pairs in the batch. The gradients of $\mathcal{L}_{\text{SACLR}}(\theta)$ can then be used in Stochastic Gradient Descent or Adam-style optimization (Kingma & Ba, 2014).

3.4 Exponential Moving Average of s

The scaling factor s can also be calculated in an expectation manner:

$$s^{-1} = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{u=1}^{2}\sum_{v=1}^{2}\left[\alpha p_{ij}^{u,v}N + (1-\alpha)\right] q_{ij}^{u,v} \quad (7)$$

$$= N^2\mathbb{E}_{i\sim\text{Uniform}(\{1,\dots,N\})}\left\{2\alpha q_{ii}^{1,2} + (1-\alpha)\mathbb{E}_{j\sim\text{Uniform}(\{1,\dots,N\})}\left\{\sum_{u=1}^{2}\sum_{v=1}^{2} q_{ij}^{u,v}\right\}\right\}. \quad (8)$$

Therefore $\xi = \frac{N^2}{B}\sum_{i\in\mathcal{B}}\left(2\alpha q_{ii}^{1,2} + (1-\alpha)\frac{1}{M}\sum_{j\in\mathcal{M}_i}\sum_{u=1}^{2}\sum_{v=1}^{2} q_{ij}^{u,v}\right)$ is the stochastic approximation of s^{-1} in a minibatch. We can then use an exponential moving average to update the estimate of $s^{-1} \leftarrow \rho s^{-1} + (1-\rho)\xi$ with a forgetting rate $\rho \in (0, 1)$ after each batch.

3.5 Approximation to Matrix Rows

We have derived SACLR with matrix-wise approximation to ψ by following SCE in the above. Next, we present the row-wise approximation to ψ, which gives a more direct connection to the cross-entropy based contrastive InfoNCE loss functions practised by the existing SimCLR and DCL (Yeh et al., 2022) unbounded by the number of negative samples.

We define $\forall a$, $\psi_{a:} = \{\psi_{ab} | b \in \{1, \dots, N\} \setminus \{a\}\}$, i.e., the a-th row of ψ with the diagonal element excluded. $\phi_{a:}$ is similarly defined over ϕ. We can then apply $D_{\text{I}}(\psi_{a:}||s_a\phi_{a:})$ to measure the discrepancy between $\psi_{a:}$ and $\phi_{a:}$ up to a scaling s_a. The contrastive learning function becomes

$$\mathcal{L}_{\text{CLR-row}}(\theta) = \sum_{a=1}^{2N} D_{\text{I}}(\psi_{a:}||s_a\phi_{a:}) \quad (9)$$

$$= \sum_{i=1}^{N}\left[-2\log q_{ii}^{1,2} + \sum_{u=1}^{2} s_{2(i-1)+u}\sum_{j=1}^{N}\sum_{v=1}^{2}\mathbb{1}_{[i\neq j \text{ or } u\neq v]} q_{ij}^{u,v}\right] + C_2 \quad (10)$$

Algorithm 1 SACLR algorithm (using matrix row approximation)

Input: Input data $\{\mathbf{x}_i\}_{i=1}^N$, weighting rate $\alpha \in [0, 1]$, forgetting rate $\rho \in (0, 1)$, number of iterations T, batch size B, number of negative samples M, and a neural network f (parameterized by θ).

Initialize the neural network f, and $s_i \leftarrow 1/N$ for $i = 1, \ldots, N$
for $t = 1, \ldots, T$ **do**
  Uniformly draw $\mathcal{B} = \{i_1, \ldots, i_B\}$ from $\{1, \ldots, N\}$
  Augment $\mathbf{x}_i$ to $\tilde{\mathbf{x}}_i^{(1)}$ and $\tilde{\mathbf{x}}_i^{(2)}$ for $i \in \mathcal{B}$
  $\mathcal{L} \leftarrow 0$
  for $i \in \mathcal{B}$ **do**
    Compute $q_{ii}^{1,2} = q\big(f(\tilde{\mathbf{x}}_i^{(1)}; \theta), f(\tilde{\mathbf{x}}_i^{(2)}; \theta)\big)$
    Uniformly draw $\mathcal{M}_i = \{j_1, \ldots, j_M\}$ from $\mathcal{B}$
    Compute $q_{ij}^{u,v} = q\big(f(\tilde{\mathbf{x}}_i^{(u)}; \theta), f(\tilde{\mathbf{x}}_j^{(v)}; \theta)\big)$ for $j \in \mathcal{M}_i$, $u \in \{1, 2\}$, and $v \in \{1, 2\}$
    $\mathcal{L} \leftarrow -2 \log q_{ii}^{1,2}$
    for $u \in \{1, 2\}$ **do**
      $\mathcal{L} \leftarrow \mathcal{L} + s_{2(i-1)+u} \frac{N}{M} \sum_{j \in \mathcal{M}_i} \sum_{u=1}^{2} \sum_{v=1}^{2} q_{ij}$
      $\xi_{2(i-1)+u} \leftarrow N(2\alpha q_{ii}^{1,2} + (1-\alpha) \frac{1}{M} \sum_{j \in \mathcal{M}_i} \sum_{v=1}^{2} q_{ij}^{u,v})$
      $s_{2(i-1)+u}^{-1} \leftarrow \rho s_{2(i-1)+u}^{-1} + (1-\rho) \xi_{2(i-1)+u}$
    end for
  end for
  Update θ with $\nabla_\theta \mathcal{L}$ using an SGD or Adam-style step.
end for

Output: Trained neural network f.

where $\mathbb{1}_{[\cdot]} = 1$ when the bracketed condition is true and otherwise 0, and $C_2 = -2N - \sum_{a=1}^{2N} \log s_a$ is a constant for fixed s_a's.

The row approximation objective is connected to SimCLR (proof in Appendix A.5):

Theorem 1. *When* $s_{2(i-1)+u}^{-1} = \sum_{jv} \mathbb{1}_{[i \neq j \text{ or } u \neq v]} q_{ij}^{u,v}, \forall i, u$, *minimizing* $\mathcal{L}_{\text{CLR-row}}(\theta)$ *is equivalent to minimizing*[1] $\mathcal{L}_{\text{SimCLR}}(\theta) = -\sum_{i=1}^{N} \sum_{u=1}^{2} \log \frac{q_{ii}^{1,2}}{\sum_{j=1}^{N} \sum_{v=1}^{2} \mathbb{1}_{[i \neq j \text{ or } u \neq v]} q_{ij}^{u,v}}$.

Differently, we employ a generalization $s_{2(i-1)+u}^{-1} = \sum_{jv} = w_{ij}^{u,v} q_{ij}^{u,v}$ with non-uniform w defined in Eq. 4. The change adaptively reduces the effect of negative pairs and leads to improvements in our experiments.

The row-wise approximation version of SACLR objective is

$$\mathcal{L}_{\text{SACLR-row}}(\theta) = \sum_{i \in \mathcal{B}} \left[-2 \log q_{ii}^{1,2} + \sum_{u=1}^{2} s_{2(i-1)+u} \frac{N}{M} \sum_{j \in \mathcal{M}_i} \sum_{v=1}^{2} q_{ij}^{u,v} \right] \tag{11}$$

The expectation form of $s_{2(i-1)+u}$ equals $N \left[\alpha q_{ii}^{1,2} + (1-\alpha) \mathbb{E}_{j \sim \text{Uniform}(\{1,\ldots,N\})} \left\{ \sum_{v=1}^{2} q_{ij}^{u,v} \right\} \right]$, and its stochastic approximation is $\xi_{2(i-1)+u} = N \left(\alpha q_{ii}^{1,2} + (1-\alpha) \frac{1}{M} \sum_{j \in \mathcal{M}_i} \sum_{v=1}^{2} q_{ij}^{u,v} \right)$. with the EMA update rule $s_{2(i-1)+u}^{-1} \leftarrow \rho s_{2(i-1)+u}^{-1} + (1-\rho) \xi_{2(i-1)+u}$.

This version of SACLR require $2N$ scaling factors. In practice, the two scaling factors of each data instance will be very similar. Therefore we can ease the requirement by only using N scaling factors in the actual implementation. The same strategy is also used by SogCLR (Yuan et al., 2022) and iSogCLR's (Qiu et al., 2023). The pseudocode of SACLR using matrix row approximation is given in Algorithm 1. The version using matrix-wise approximation is similar and given in Appendix A.6.

[1] The original SimCLR use exponential over cosine similarities. They are equivalent to the Gaussian kernels given that $\tilde{\mathbf{y}}_i^{(1)}$'s and $\tilde{\mathbf{y}}_i^{(2)}$'s are normalized.

Table 1: Top-1 linear validation accuracy on ImageNet100. * marks improved results reported by Qiu et al. (2023). Standard deviations are from three different runs.

Method	Batchsize	400EP
SACLR-1 matrix (ours)	128	**82.30** ($\pm$ 0.46)
SACLR-1 row (ours)	128	81.59 ($\pm$ 0.07)
SimCLR (Chen et al., 2020a) *	256	79.96 ($\pm$ 0.2)
SogCLR (Yuan et al., 2022) *	256	80.54 ($\pm$ 0.14)
iSogCLR (Qiu et al., 2023) *	256	81.14 ($\pm$ 0.19)

Table 2: Top-1 linear validation accuracy on ImageNet1k. Standard deviations are from three different runs.

Method	Batchsize	100EP	400EP
SACLR-1 matrix (ours)	128	**64.94** $\pm$ (0.16)	**67.57** ($\pm$ 0.01)
SACLR-1 row (ours)	128	64.78 ($\pm$ 0.05)	67.50 ($\pm$ 0.14)
SimCLR (Chen et al., 2020a)	256	62.80	65.70
SogCLR (Yuan et al., 2022)	128	64.90	67.40

4 EXPERIMENTS

We employ our methods on color images from ImageNet (Russakovsky et al., 2015) and CIFAR (Alex, 2009). The image augmentations and network architecture on ImageNet follows SimCLR (Chen et al., 2020a), LARS-optimizer (You et al., 2017) and the projector from VIC-REG (Bardes et al., 2021). Evaluation follows the linear evaluation protocol on frozen representations from the backbone. Full implementation details on ImageNet are described in Appendix A.1 and on CIFAR in Appendix A.2. While the introduced algorithms can work with $M > 1$ negative samples we focus on SACLR with $M = 1$ and use $M \gg 1$ for ablations.

ImageNet: We find that our methods achieve good results from image SSL-pretraining on a merited image-dataset as ImageNet. The results show that our methods can consistently have improvements to many existing CL-approaches in the low batchsize setting and in addition can be more memory efficient. On ImageNet we compare our methods to existing CL approaches such as SimCLR (Chen et al., 2020a), and recent stochastic estimation based CL approaches SogCLR (Yuan et al., 2022) and iSogCLR (Qiu et al., 2023). All methods are pretrained for different amounts of epochs and each reported value is from one run from scratch. The linear classification accuracies after image-pretraining on ImageNet100 (Wu et al., 2019) are presented in Table 1 and the results on ImageNet1k (Russakovsky et al., 2015) are presented in Table 2. We exclusively report values from each methods respective paper unless explicitly mentioned. The full results from image pretraining on ImageNet1k and ImageNet100 are presented in Table 5 and Table 6 in Appendix. Remarkably we see that with $M = 1$ negative samples per anchor SACLR can give performance improvements to SimCLR and SogCLR-variants which use $M \gg 1$ on both ImageNet1k and ImageNet100. We prove that our methods are also very effective for unsupervised cluster visualization in Figure 2.

CIFAR: We also include results with a ResNet18 encoder on CIFAR and Imagenette (10 class subset of ImageNet; Howard, 2019) in Tables 3 and 4. We pretrain for 1000 epochs on CIFAR and 800 epochs on Imagenette. On CIFAR the classifer is a linear layer and we include reported values of SimCLR and SimSiam from solo-learn da Costa et al. (2022). On Imagenette we use 20NN classifier from scikit-learn (Pedregosa et al., 2011) with cosine-weighting $w = \exp(\text{CosSim}(\mathbf{a}, \mathbf{b})/0.07)$ (Wu et al., 2018; Caron et al., 2021; Balestriero et al., 2023) and find reported values from Lightly AI [2].

Whereas existing approaches in CL typically are advised to find $M \gg 1$ negative samples per anchor (Chen et al., 2020a; He et al., 2020; Yeh et al., 2022; Damrich et al., 2023) or require additional variables to network parameters which scale with the number of input-instances N (Yuan et al., 2022; Qiu et al., 2023), our methods can leverage $M = 1$ negative sample per anchor and a single additional variable devoted for the stochastic estimation. A more comprehensive overview of these

[2] https://docs.lightly.ai/self-supervised-learning/getting_started/benchmarks.html

Table 3: Top 1 accuracy with a linear classifier on CIFAR. Results marked with * are from solo-learn (da Costa et al., 2022) and ** are from Qiu et al. (2023). Standard deviations are from three different runs.

Method	CIFAR10	CIFAR100
SACLR-1 matrix (ours)	**93.07** ($\pm$ 0.17)	70.22 ($\pm$ 0.38)
SACLR-1 row (ours)	92.86 ($\pm$ 0.08)	70.39 ($\pm$ 0.3)
SACLR-all matrix (ours)	92.91	70.89
SACLR-all row (ours)	92.98	**71.50**
SimCLR (Chen et al., 2020a) *	90.74	65.78
SogCLR (Yuan et al., 2022)**	90.07 ($\pm$ 0.10)	65.18 ($\pm$ 0.10)
iSogCLR (Qiu et al., 2023)**	90.25 ($\pm$ 0.09)	65.95 ($\pm$ 0.07)
SimSiam (Chen & He, 2021) *	90.51	66.04
ReSSL (Zheng et al., 2021) *	90.63	65.92

Table 4: Top-1 accuracy using a 20NN-classifier for CIFAR and Imagenette datasets.

Method	CIFAR10	CIFAR100	Imagenette
SACLR-1 matrix (ours)	91.65 ($\pm$ 0.01)	67.02 ($\pm$ 0.21)	**90.74** ($\pm$ 0.04)
SACLR-1 row (ours)	91.75 ($\pm$ 0.06)	67.33 ($\pm$ 0.03)	90.68 ($\pm$ 0.2)
SACLR-all matrix (ours)	91.83	67.89	-
SACLR-all row (ours)	**91.87**	68.05	-
SimCLR (Chen et al., 2020a)	90.59 *	65.32 *	88.90
SogCLR (Yuan et al., 2022)	89.98	65.95	-
iSogCLR (Qiu et al., 2023)	91.69	**68.85**	-
SimSiam (Chen & He, 2021)	90.83 *	66.43 *	87.20
ReSSL (Zheng et al., 2021)	90.80 *	66.08 *	-

memory requirements are demonstrated in Table 7 in Appendix. More tables of runtime and memory usage in Appendix A.3 are found in Table 8, 9 and 10.

5 Ablation studies

We perform ablations to see the effect of different forgetting rates, weighting rates, global compared to individualized adaptive scaling parameters and M negative samples per anchor in Appendix A.4. The findings are that our methods are robust against hyperparameters and fewer negative samples.

This work presented a matrix-method with a one global adaptive scaling factor and a row-method with individualized adaptive scaling factors. We study the impact of the respective methods in Table 14 in Appendix. Surprisingly we find that the matrix-method performs better or evenly to the row-method which has a cost of more variables. Visual example is shown in Figure 3 where we log the positive and negative values with the estimated partition function(s) after each epoch. While the row-method can offer a more specific estimate especially in the early stages it might seem to converge to a similar value in the later stages. We also observe a negligible difference between SACLR with $M = 1$ negative sample compared to the full batch mode version where $M = 2 \times 128 - 2$.

The hyperparameters of most importance are the forgetting rate ρ and weighting rate α, in addition to initialization of scaling factors. We tune the forgetting rate ρ on Imagenette in Table 12 and Table 13 in Appendix. We find that $\rho = 0.99$ works best for the matrix-method and $\rho = 0.9$ for the row-method. The matrix-method re-estimates the scaling factor after each minibatch which is contrary to the row-method which re-estimates per instance approximately at each epoch, see Alg.(1) and Alg.(2) for more details. A consequence of this can be a necessity to use a higher forgetting rate, which is demonstrated in Table 13 with a linear classifier. The weighting rate α is ablated in Table 15 on ImageNet for SACLR-1. We initially set $\alpha = 0.5$ and see that setting a lower weighting-rate $\alpha = 0.125$ gives better performance.

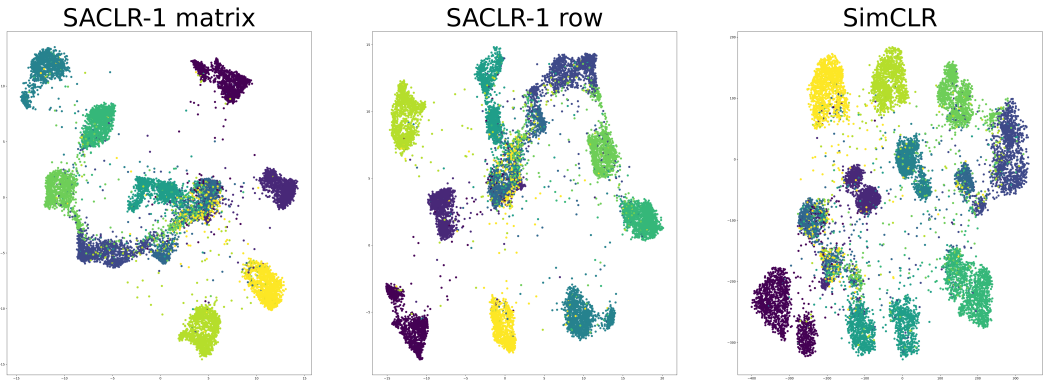
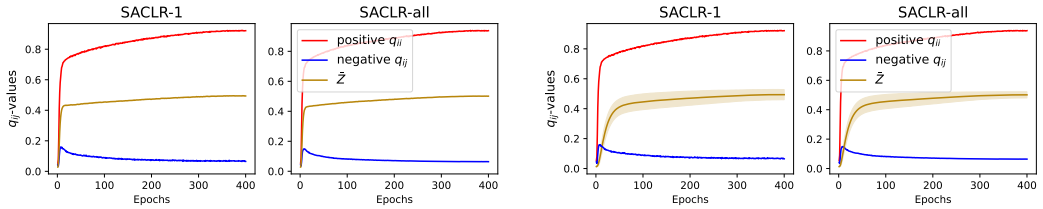


Figure 2: Embedding visualizations of Imagenette dataset. All compared methods used the Cauchy kernel and were trained with raw images. We followed the strategies by Böhm et al. (2023): we first pretrained the models in 8192-dimensional space over 800 epochs and then finetuned a 2D projector over 250 epochs. SACLR-1 used $M = 1$ negative sample while SimCLR used $M = 128 \times 2 - 2$ for each image in a batch.



(a) **Matrix-method** (b) **Row-method**

Figure 3: Optimization dynamics: positive and negative values with estimated partition function(s) on Imagenette. We include the variability for the row method with standard deviations.

6 Conclusion and Future Work

We have approached contrastive learning as a matrix approximation problem. A key component of our method is an adaptive scaling factor, which optimizes separable similarities and enables memory-efficient stochastic approximation algorithms. This allows our approach to require significantly smaller training batches and as few as one negative pair, making it substantially more efficient than other contrastive learning techniques. Despite its economical design, our method has demonstrated the effectiveness of SACLR on CIFAR and ImageNet classification tasks, with consistent performance improvements to the compared methods.

In the future, our work could be extended in several promising directions. One potential area of improvement is integrating contextual learning within individual data instances, in addition to the contrastive learning across data populations. By capturing more detailed relationships within a single data sample, such as spatial or sequential patterns, we could significantly enhance the model's ability to learn richer, more nuanced representations. This integration could improve performance in tasks that require fine-grained understanding, like object detection or temporal data analysis.

Another direction is replacing the traditional convolutional neural network with more advanced architectures, such as self-attention-based models like the Transformer. Self-attention mechanisms have shown remarkable success in capturing long-range dependencies and global patterns in various domains. Incorporating this into our framework could lead to more expressive models that can handle more complex data types, such as video or multi-modal inputs, while further improving efficiency and performance.

REPRODUCIBILITY STATEMENT

The code is made available and the README includes examples of how to reproduce the results. In addition we provide description of data processing and implementation details in A.1 and A.2.

REFERENCES

Krizhevsky Alex. Learning multiple layers of features from tiny images. *https://www. cs. toronto. edu/kriz/learning-features-2009-TR. pdf*, 2009.

Ehsan Amid and Manfred K Warmuth. Trimap: Large-scale dimensionality reduction using triplets. *arXiv preprint arXiv:1910.00204*, 2019.

Randall Balestriero and Yann LeCun. Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 26671–26685. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/aa56c74513a5e35768a11f4e82dd7ffb-Paper-Conference.pdf.

Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, et al. A cookbook of self-supervised learning. *arXiv preprint arXiv:2304.12210*, 2023.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Jan Niklas Böhm, Philipp Berens, and Dmitry Kobak. Unsupervised visualization of image datasets using contrastive learning. In *International Conference on Learning Representations*, 2023.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In J. Cowan, G. Tesauro, and J. Alspector (eds.), *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann, 1993. URL https://proceedings.neurips.cc/paper_files/paper/1993/file/288cc0ff022877bd3df94bc9360b9c5d-Paper.pdf.

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

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

Changyou Chen, Jianyi Zhang, Yi Xu, Liqun Chen, Jiali Duan, Yiran Chen, Son Tran, Belinda Zeng, and Trishul Chilimbi. Why do we need large batchsizes in contrastive learning? a gradient-bias perspective. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 33860–33875. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/db174d373133dcc6bf83bc98e4b681f8-Paper-Conference.pdf.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020a.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020b.

Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pp. 539–546. IEEE, 2005.

Victor Guilherme Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. solo-learn: A library of self-supervised methods for visual representation learning. *Journal of Machine Learning Research*, 23(56):1–6, 2022. URL http://jmlr.org/papers/v23/21-1155.html.

Sebastian Damrich, Jan Niklas Böhm, Fred A Hamprecht, and Dmitry Kobak. From t-SNE to UMAP with contrastive learning. In *International Conference on Learning Representations*, 2023.

Virginia de Sa. Learning classification with unlabeled data. In J. Cowan, G. Tesauro, and J. Alspector (eds.), *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann, 1993. URL https://proceedings.neurips.cc/paper_files/paper/1993/file/e0ec453e28e061cc58ac43f91dc2f3f0-Paper.pdf.

Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3015–3024. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/ermolov21a.html.

Quentin Garrido, Yubei Chen, Adrien Bardes, Laurent Najman, and Yann LeCun. On the duality between contrastive and non-contrastive self-supervised learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=kDEL91Dufpa.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *Proceedings of the ieee/cvf International Conference on computer vision*, pp. 6391–6400, 2019.

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*, 33:21271–21284, 2020.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/gutmann10a.html.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 2, pp. 1735–1742. IEEE, 2006.

Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *Advances in Neural Information Processing Systems*, 34:5000–5011, 2021.

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

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

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.

Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. In S. Becker, S. Thrun, and K. Obermayer (eds.), *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2002. URL https://proceedings.neurips.cc/paper_files/paper/2002/file/6150ccc6069bea6b5716254057a194ef-Paper.pdf.

Jeremy Howard. Imagenette: A smaller subset of 10 easily classified classes from imagenet, March 2019. URL https://github.com/fastai/imagenette.

Tianyang Hu, Zhili LIU, Fengwei Zhou, Wenjia Wang, and Weiran Huang. Your contrastive learning is secretly doing stochastic neighbor embedding. In *International Conference on Learning Representations*, 2023.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *Advances in neural information processing systems*, 33: 21798–21809, 2020.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):857–876, 2023. doi: 10.1109/TKDE.2021.3090866.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6707–6717, 2020.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Zi-Hao Qiu, Quanqi Hu, Zhuoning Yuan, Denny Zhou, Lijun Zhang, and Tianbao Yang. Not all semantics are created equal: Contrastive self-supervised learning with automatic temperature individualization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28389–28421. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/qiu23a.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*, 2020.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

Anshul Shah, Suvrit Sra, Rama Chellappa, and Anoop Cherian. Max-margin contrastive learning, 2021. URL https://arxiv.org/abs/2112.11450.

Rohan Sharma, Kaiyi Ji, Changyou Chen, et al. Auc-cl: A batchsize-robust framework for self-supervised contrastive representation learning. In *The Twelfth International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=YgMdDQB09U.

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/6b180037abbebea991d8b1232f8a8ca9-Paper.pdf.

Yi Sui, Tongzi Wu, Jesse C. Cresswell, Ga Wu, George Stein, Xiao Shi Huang, Xiaochen Zhang, and Maksims Volkovs. Self-supervised representation learning from random data projectors. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=EpYnZpDpsQ.

Zhiquan Tan, Yifan Zhang, Jingqin Yang, and Yang Yuan. Contrastive learning is spectral clustering on similarity graph. In *The Twelfth International Conference on Learning Representations*, 2024.

Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. Visualizing large-scale and high-dimensional data. In *Proceedings of the 25th international conference on world wide web*, pp. 287–297, 2016.

Yuandong Tian. Understanding deep contrastive learning via coordinate-wise optimization. *Advances in Neural Information Processing Systems*, 35:19511–19522, 2022.

Nenad Tomasev, Ioana Bica, Brian McWilliams, Lars Buesing, Razvan Pascanu, Charles Blundell, and Jovana Mitrovic. Pushing the limits of self-supervised resnets: Can we outperform supervised learning without labels on imagenet? *arXiv preprint arXiv:2201.05119*, 2022.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *Journal of Machine Learning Research*, 22(201):1–73, 2021. URL http://jmlr.org/papers/v22/20-1061.html.

Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of machine learning research*, 10(2), 2009.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 374–382, 2019.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3733–3742, 2018.

Zhirong Yang, Yuwei Chen, Denis Sedov, Samuel Kaski, and Jukka Corander. Stochastic cluster embedding. *Statistics and Computing*, 33(1):12, 2023.

Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. In *European conference on computer vision*, pp. 668–684. Springer, 2022.

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

Zhuoning Yuan, Yuexin Wu, Zihao Qiu, Xianzhi Du, Lijun Zhang, Denny Zhou, and Tianbao Yang. Provable stochastic optimization for global contrastive learning: Small batch does not harm performance. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 25760–25782. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/yuan22b.html.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*, pp. 12310–12320. PMLR, 2021.

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

Mingkai Zheng, Shan You, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Ressl: Relational self-supervised learning with weak augmentation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 2543–2555. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/14c4f36143b4b09cbc320d7c95a50ee7-Paper.pdf.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *International Conference on Learning Representations (ICLR)*, 2022.

A APPENDIX

A.1 IMPLEMENTATION DETAILS IMAGENET

A.1.1 DATASETS AND AUGMENTATIONS

ImageNet is an eminent image-dataset and contains color images from rich range of different classes. This work use the subsets ImageNet1k (Russakovsky et al., 2015) (ILSVRC2012) which contains 1,281,167 training images from 1000 different classes, and ImageNet100 (Wu et al., 2019) which contains 126,689 training images from 100 different classes. The pre-defined training and validation split are used, and we evaluate our methods on the validation set, similar to existing work. For ablations and hyperparameter tuning shorter runs and the image-dataset Imagenette (Howard, 2019) are used. Imagenette is a subset from ImageNet with 10 different classes. All images from ImageNet are resized to two (224×224) image views without any multi-crop from Caron et al. (2020). All images are applied the augmentations from SimCLR (Chen et al., 2020a) during pretraining, and we use the augmentation implementations from torchvision. We download ImageNet (ILSVRC2012) from the official website and apart from that use the built-in datasets to torchvision.

A.1.2 ARCHITECTURE

The neural network architecture in this work adhere to SimCLR (Chen et al., 2020a), and contains a backbone encoder and a projector. The backbone encoder is a ResNet50 (He et al., 2016) and the projector is the 3-layered MLP from VIC-REG (Bardes et al., 2021). The projector is applied on the output of the ResNet's avg-pooling layer and the output of each linear layer is 8192-d. We use the ResNet architecture from torchvision.

A.1.3 OPTIMIZATION

The neural network optimizer in this work is the LARS-optimizer (You et al., 2017) with the square-root learning rate scaling for low batchsizes proposed by SimCLR (Chen et al., 2020a) $lr = \sqrt{batchsize} \times 0.075$, and weight decay $wd = 10^{-4}$. The learning rate is linearly warmed up the 10 first epochs and then annealed with a cosine schedule (Loshchilov & Hutter, 2016) until it reaches $lr/1000$. The default batchsize is 128.

A.1.4 LINEAR EVALUATION

Evaluation follows the linear classifier protocol on frozen backbone representations (Zhang et al., 2016; Balestriero et al., 2023). This work keep the optimization settings from Caron et al. (2020); Zbontar et al. (2021) with SGD-optimizer, momentum $= 0.9$, $wd = 10^{-6}$ and $lr = 0.3$. The learning rate is annealed with a cosine-scheduler down to zero over 100 epochs. During training we use random-cropping, -resizing and horizontal flipping, and we test with a center crop.

A.1.5 LOSS FUNCTION HYPERPARAMETERS

This work set by default weighting rate $\alpha = 0.125$. The forgetting rate is tuned $\rho \in \{0.9, 0.99, 0.999\}$ on Imagenette. The adaptive scaling parameter(s) are initialized such that the partition function(s) at start correspond to 0.01, e.g. $s = N^{-2}10^2$ where N is the number of training images.

A.1.6 SIMILARITY FUNCTION

The similarity function in this work is a squared exponential kernel

$$q\big(\mathbf{y}_i, \mathbf{y}_j\big) = \exp(-\|\overline{\mathbf{y}}_i - \overline{\mathbf{y}}_j\|^2/2\tau^2) \tag{12}$$

on the unit-sphere normalized neural network output $\overline{\mathbf{y}_i} = \mathbf{y}_i/\|\mathbf{y}_i\|$ and $\overline{\mathbf{y}_j} = \mathbf{y}_j/\|\mathbf{y}_j\|$, with a chosen temperature $\tau \in \mathbb{R}_+$. In this work we set $\tau = 0.5$.

Table 5: **ImageNet100:** Top 1 linear validation accuracy on ImageNet100 with ResNet50. * is a mark of improved results reported by Qiu et al. (2023). Standard deviations are from three different runs.

Method	Batchsize	400EP
SACLR-1 matrix (ours)	128	82.30 ($\pm$ 0.46)
SACLR-1 row (ours)	128	81.59 ($\pm$ 0.07)
SACLR-all matrix (ours)	128	82.50
SACLR-all row (ours)	128	82.18
SimCLR (Chen et al., 2020a) *	256	79.96 ($\pm$ 0.2)
SogCLR (Yuan et al., 2022) *	256	80.54 ($\pm$ 0.14)
iSogCLR (Qiu et al., 2023) *	256	81.14 ($\pm$ 0.19)

Table 6: **ImageNet1k:** Top 1 linear validation accuracy on ImageNet1k with ResNet50. Standard deviations are from three different runs.

Method	Batchsize	100EP	400EP
SACLR-1 matrix (ours)	128	64.94 ($\pm$ 0.16)	67.57 ($\pm$ 0.01)
SACLR-1 row (ours)	128	64.78 ($\pm$ 0.05)	67.50 ($\pm$ 0.14)
SACLR-all row (ours)	128	65.90	67.34
SACLR-1 row (ours)	256	65.35	-
SACLR-all row (ours)	256	66.75	-
SimCLR (Chen et al., 2020a)	256	62.80	65.70
SogCLR (Yuan et al., 2022)	128	64.90	67.40

A.1.7 Compute requirements

This work use exclusively one GPU setups with at least 24 GB VRAM. On ImageNet we pretrain with Nvidia H100 and Nvidia A100 GPUs on a SLURM cluster, and use a Nvidia RTX4090 for other datasets.

A.2 Implementation details CIFAR

CIFAR (Alex, 2009) contains downsampled (32×32) color images with 10 different classes in CIFAR10 and 100 different classes in CIFAR100. The experimental setup on CIFAR is more or less the same to the setup on ImageNet in A.1 but add a few alternations to downsampled (32×32) images following Chen et al. (2020a); Chen & He (2021). The ResNet-architecture and image augmentations are adapted to CIFAR (Chen et al., 2020a). The optimization settings for unsupervised pretraining and linear frozen evaluation are the same to Chen & He (2021) on CIFAR10.

A.3 Additional results

We compare our methods to existing CL methods as SimCLR (Chen et al., 2020a) and its recent improvements SogCLR (Yuan et al., 2022) and iSogCLR (Qiu et al., 2023) for low batchsize settings. All methods use ResNet50 backbone and augmentation strategies from Chen et al. (2020a) on two (224×224) image views. The full results from the linear classifier after image-pretraining on ImageNet100 are shown in Table 5 and on ImageNet1k in Table 6. Results of SACLR use $\alpha = 0.125$. We provide an overview of computational costs and additional memory requirements for our own methods and existing stochastic estimation based CL-approaches in Table 7, 8, 9 and 10.

A.4 Ablation results

We continue from Section 5 with more details of investigations into effects of number of negative samples, approximation-methods and choice of hyperparameters. By default we set $\alpha = 0.5$ for these trials. The forgetting rate is tuned $\rho \in \{0.9, 0.99, 0.999\}$ on Imagenette (Howard, 2019), where we pretrain for 400 epochs and use a separate validation set split (20%) from the training set.

Table 7: **Memory complexity and additional memory complexity to network parameters**. We compare memory efficiency and required extra variables for the stochastic estimation between CL methods. We denote the number of input instances by N, negative samples per anchor M and batchsize B.

Method	M	Additional variables
SACLR-1 matrix-method (ours)	1	1
SACLR-1 row-method (ours)	1	N
SACLR-all matrix-method (ours)	$2B$ - 2	1
SACLR-all row-method (ours)	$2B$ - 2	N
SimCLR (Chen et al., 2020a)	$2B$ - 2	0
SogCLR (Yuan et al., 2022)	$2B$ - 2	N
iSogCLR (Qiu et al., 2023)	$2B$ - 2	$4N$

Table 8: **Computational costs on CIFAR**. GPU peak memory usage over large batch training over ranges B often used with the LARS (You et al., 2017) optimizer. In this study the network is a ResNet18 on CIFAR10 over 10EP. The last fc-layer has 128 units. We downloaded the PyTorch implementations from SogCLR (Yuan et al., 2022) and iSogCLR (Qiu et al., 2023) in this study. Experiments conducted on a 10 core CPU with 32GB RAM and NVIDIA H100 80GB VRAM GPU. 1K denote 1024 and values over 80GB are OOM-errors.

Method $\setminus B$	128	1K	2K	4K	8K	16K	32K
SACLR-1	1.17 GB	2.69 GB	4.38 GB	7.59 GB	14.35 GB	27.12 GB	53.17 GB
SACLR-all	1.17 GB	2.73 GB	4.58 GB	8.50 GB	17.73 GB	39.44 GB	$>$ 80GB
SimCLR	1.17 GB	2.69 GB	4.43 GB	7.94 GB	15.48 GB	30.42 GB	$>$ 80GB
SogCLR	1.17 GB	2.78 GB	4.62 GB	8.74 GB	17.70 GB	50.92 GB	$>$ 80GB
iSogCLR	1.17 GB	2.73 GB	4.50 GB	7.99 GB	14.72 GB	38.92 GB	$>$ 80GB

We report accuracy with the linear classifier in Table 12 and Table 13. The impact of applying row-based estimation compared to matrix-based estimation is studied in ImageNet1k over 100 epoch runs and ImageNet100 with a linear classifier in Table 14. Notably we see that the number of negative samples do not leave a significant impact on SACLR from Table 12, Table 13 and Table 14. The weighting term is ablated with a linear classifier over shorter 100 epoch runs on ImageNet1k and ImageNet100 in Table 15. We see that a lower weighting term $\alpha < 0.5$ can be better for SACLR-1.

Table 9: **Computational costs on MNIST**. GPU peak memory usage from large batch training settings often used with the LARS (You et al., 2017) optimizer on MNIST with different batchsize B. Now we use only a linear layer with 128 units to isolate the impact of loss functions as much as possible. We downloaded the PyTorch implementations from SogCLR (Yuan et al., 2022) and iSogCLR (Qiu et al., 2023) in this study. We used a NVIDIA H100 80GB GPU and 10 core CPU with 32GB. All values over 80GB are OOM-errors.

Method $\backslash B$	128	1K	2K	4K	8K	16K	32K
SACLR-1	0.80 GB	0.82 GB	0.84 GB	0.86 GB	0.90 GB	1.03 GB	1.37 GB
SACLR-all	0.80 GB	0.92 GB	1.20 GB	2.31 GB	6.85 GB	24.93 GB	>80GB
SimCLR	0.80 GB	0.88 GB	1.05 GB	1.75 GB	4.58 GB	15.89 GB	61.02 GB
SogCLR	0.80 GB	0.97 GB	1.38 GB	3.06 GB	9.83 GB	36.90 GB	>80GB
iSogCLR	0.80 GB	0.90 GB	1.18 GB	2.31 GB	6.84 GB	24.89 GB	>80GB

Table 10: **Runtime and computational costs**. Time complexity and peak memory usage during 100 epoch training on CIFAR10 with the highest batchsize possible 16384 for a NVIDIA H100 80GB GPU without any OOM error. Here M is the number of negative samples per image in batch. We used the respective PyTorch implementations from SogCLR (Yuan et al., 2022) and iSogCLR (Qiu et al., 2023) in this study. The network is a ResNet18.

Method	Time / 100 epochs	Peak GPU memory	M
SACLR-1 matrix-method (ours)	0.44h	27.12 GB	1
SACLR-1 row-method (ours)	0.44h	27.12 GB	1
SACLR-all matrix-method (ours)	0.43h	39.44 GB	16384×2 - 2
SACLR-all row-method (ours)	0.44h	39.44 GB	16384×2 - 2
SimCLR (Chen et al., 2020a)	0.43h	30.42 GB	16384×2 - 2
SogCLR (Yuan et al., 2022)	0.50h	50.92 GB	16384×2 - 2
iSogCLR (Qiu et al., 2023)	0.48h	38.92 GB	16384×2 - 2

A.5 Proof of Theorem 1

Proof. When $s^{-1}_{2(i-1)+u} = \sum_{j=1}^{N}\sum_{v=1}^{2} \mathbb{1}_{[i\neq j \text{ or } u\neq v]} q_{ij}^{u,v}, \forall i, u$

$$\mathcal{L}_{\text{CLR-row}}(\theta) = \sum_{i=1}^{N}\left[-2\log q_{ii}^{1,2} + \sum_{u=1}^{2} s_{2(i-1)+u} \sum_{j=1}^{N}\sum_{v=1}^{2} \mathbb{1}_{[i\neq j \text{ or } u\neq v]} q_{ij}^{u,v}\right] - 2N - \sum_{a=1}^{2N}\log s_a \tag{13}$$

$$= \left(\sum_{i=1}^{N} -2\log q_{ii}^{1,2}\right) + 2N - 2N - \sum_{a=1}^{2N}\log s_a \tag{14}$$

$$= -\sum_{i=1}^{N}\sum_{u=1}^{2}\log q_{ii}^{1,2} + \sum_{i=1}^{N}\sum_{u=1}^{2}\log \sum_{j=1}^{N}\sum_{v=1}^{2} \mathbb{1}_{[i\neq j \text{ or } u\neq v]} q_{ij}^{u,v} \tag{15}$$

$$= -\sum_{i=1}^{N}\sum_{u=1}^{2}\log \frac{\log q_{ii}^{1,2}}{\sum_{j=1}^{N}\sum_{v=1}^{2} \mathbb{1}_{[i\neq j \text{ or } u\neq v]} q_{ij}^{u,v}} \tag{16}$$

□

Table 11: **Time complexity** Time complexity of updating s on CIFAR100 over 100 epochs for SACLR-1 matrix. Each run were on a RTX 4080 12GB.

	Normal	Constant
Time / 100EP	0.67 h	0.67 h

Table 12: **Row-method impact of forgetting rates ρ and M negative samples per anchor.** Linear classification accuracy on Imagenette after 400 epochs pretraining with ResNet50 and batchsize $B = 128$. We set $\alpha = 0.5$ for each method. We report accuracy on a separate validation set split from the training set.

	$M = 1$			$M = 2 \times 128 - 2$		
ρ	0.9	0.99	0.999	0.9	0.99	0.999
	89.334	89.281	85.480	**90.337**	90.126	89.229

Table 13: **Matrix-method impact of forgetting rates ρ and M negative samples per anchor.** Linear classification accuracy on Imagenette after 400 epochs pretraining with ResNet50 and batchsize $B = 128$. We set $\alpha = 0.5$ for each method. We report accuracy on a separate validation set split from the training set.

	$M = 1$			$M = 2 \times 128 - 2$		
ρ	0.9	0.99	0.999	0.9	0.99	0.999
	89.493	**89.915**	89.704	90.443	**90.549**	90.285

A.6 ALGORTIHMS

Table 14: **Impact of row-based estimation method compared to matrix-based estimation**. Linear classification accuracy from pretraining with batchsize $B = 128$ and M negative samples per anchor, where we set by default $\alpha = 0.5$. We pretrain on ImageNet1k for 100 epochs and for 400 epochs on ImageNet100.

	$M = 1$		$M = 2 \times 128 - 2$	
	Row-method	Matrix-method	Row-method	Matrix-method
ImageNet100 (400EP)	81.240	81.640	81.880	81.100
ImageNet1k (100EP)	63.638	63.560	64.053	63.594

Table 15: **Impact of weighting term α.** Ablation studies impact of weighting term on SACLR with $M = 1$ negative samples per anchor.

	Row-method		Matrix-method	
	0.125	0.5	0.125	0.5
ImageNet100 (400Ep)	**81.70**	81.24	**82.67**	81.64
ImageNet1k (100Ep)	**64.71**	63.63	**64.75**	63.56

Algorithm 2 SACLR algorithm (using matrix-wise approximation)

Input: Input data $\{\mathbf{x}_i\}_{i=1}^N$, weighting rate $\alpha \in [0, 1]$, forgetting rate $\rho \in (0, 1)$, number of iterations T, batchsize B, number of negative samples M, a neural network f with parameters θ.

Initialize the neural network f and $s = 1/N^2$
for $t = 1, \ldots, T$ **do**
  Uniformly draw $\mathcal{B} = \{i_1, \ldots, i_B\} \subset \{1, \ldots, N\}$
  Augment $\mathbf{x}_i$ into $\tilde{\mathbf{x}}_i^{(1)}$ and $\tilde{\mathbf{x}}_i^{(2)}$, $\forall i \in \mathcal{B}$
  Compute positive pairs $\{q_{ii}^{1,2} = q\big(f(\tilde{\mathbf{x}}_i^{(1)}; \theta), f(\tilde{\mathbf{x}}_i^{(2)}; \theta)\big)\}$ for each $i \in \mathcal{B}$
  $\mathcal{L} \leftarrow 0$, $\xi \leftarrow 0$
  for $i \in \mathcal{B}$ **do**
    Uniformly draw $\mathcal{M}_i = \{j_1, \ldots, j_M\} \subseteq \mathcal{B}$
    Compute negative pairs $q_{ij}^{u,v} = q\big(f(\tilde{\mathbf{x}}_i^{(u)}; \theta), f(\tilde{\mathbf{x}}_j^{(v)}; \theta)\big)$ $\forall j \in \mathcal{M}_i$, $u, v \in \{1, 2\}$
    $\mathcal{L} \leftarrow \mathcal{L} - 2 \log q_{ii}^{1,2} + s \frac{N}{M} \sum_{j \in \mathcal{M}_i} \sum_{u=1}^{2} \sum_{v=1}^{2} q_{ij}^{u,v}$
    $\xi \leftarrow \xi + \frac{N^2}{B} \sum_{i \in \mathcal{B}} \Big(2\alpha q_{ii}^{1,2} + (1-\alpha) \frac{1}{M} \sum_{j \in \mathcal{M}_i} \sum_{u=1}^{2} \sum_{v=1}^{2} q_{ij}^{u,v}\Big)$
  end for
  $s^{-1} \leftarrow \rho s^{-1} + (1 - \rho)\xi$
  Update θ with $\nabla_\theta \mathcal{L}$ using an SGD or Adam-style step.
end for

Output: Trained neural network f.