SigCLR: Sigmoid Contrastive Learning of Visual Representations

Ömer Veysel Çağatan
Department of Computer Engineering
Koç University
Sarıyer, İstanbul 34450
ocagatan19@ku.edu.tr

Abstract

We propose SigCLR: Sigmoid Contrastive Learning of Visual Representations. SigCLR utilizes the logistic loss that only operates on pairs and does not require a global view as in the cross-entropy loss used in SimCLR. We show that logistic loss shows competitive performance on CIFAR-10, CIFAR-100, and Tiny-IN compared to other established SSL objectives. Our findings verify the importance of learnable bias as in the case of SigLUP, however, it requires a fixed temperature as in the SimCLR to excel. Overall, SigCLR is a promising replacement for the SimCLR which is ubiquitous and has shown tremendous success in various domains.

1 Introduction

Contrastive learning has been widely adopted as a pretraining objective in vision tasks to develop universal vision backbones, addressing the scalability challenges of supervised learning due to the high labeling costs [7, 8, 12, 18, 24, 27, 41]. This approach not only eliminates the need for labeled data but also allows models to be trained on large-scale image datasets [29], aiming to replicate the success seen in the language domain [36, 28]. However, a significant limitation of contrastive learning is its reliance on negative samples, which necessitates large batch sizes [7]. In contrast, non-contrastive methods [1, 4, 10, 13, 16, 30, 44, 5] focus solely on positive pairs and tend to perform well, particularly in scenarios where smaller batch sizes are used.

Although non-contrastive objectives have a clear advantage over contrastive objectives in vision pretraining, their usage is limited [34, 46, 47]. On the other hand contrastive learning has been extensively used and shown to excel as a domain-agnostic representation learning objective. This includes language [14, 15, 35, 31, 26], language-image [32, 45, 23], graphs [43], biology [38], chemistry [33] and medicine [39, 6].

SSL models that are cast under contrastive learning usually modify InfoNCE objective[37] which can also be described as softmax contrastive loss. A simple alternative to softmax contrastive loss is to use sigmoid loss and treat every pair as a binary classification problem. Even though it has a very straightforward recipe, it has been shown to be inferior to softmax objective in image pretraining [7, 20]. SigLIP Zhai et al. [45] revisits the sigmoid objective and demonstrates that it is a more efficient and equally performant model as the CLIP [32] which employs the softmax contrastive objective. The central innovation in the SigLIP [45] lies in its adoption of learnable temperature and bias. This stands in contrast to the softmax approach, which relies solely on a fixed temperature in its loss formulation.

An important consideration is whether the success of SigLIP can be applied to vision pretraining. With this in mind, we revisit the sigmoid loss in the context of vision pretraining, where it falls short compared to softmax. To explore this further, we introduce SigCLR: Sigmoid Contrastive Learning

NeurIPS 2024 Workshop: Self-Supervised Learning - Theory and Practice.

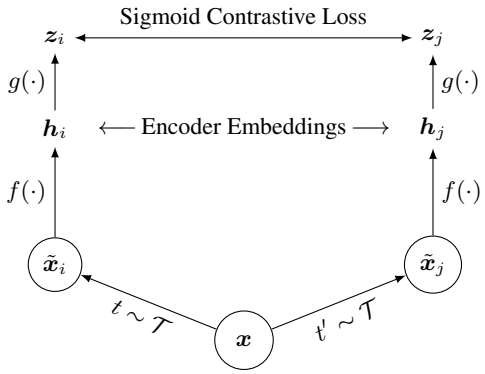

Figure 1: SigCLR follows a highly similar setup as SimCLR [7]. We randomly select two distinct data augmentation operators, denoted as $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$, from the same family of augmentations from [30]. These operators are applied independently to each data example, resulting in two correlated views. The training process involves optimizing a base encoder network, denoted as $f(\cdot)$, and a projection head, denoted as $g(\cdot)$, to maximize agreement between the representations produced by the augmented views. This optimization is achieved through the utilization of a sigmoid contrastive loss. Upon completion of training, the projection head is discarded and only encoder embeddings are utilized for evaluations.

of Visual Representations to investigate the potential of sigmoid-based contrastive pretraining for vision.

Our findings are consistent with SigLIP which emphasizes the significance of learnable bias for the sigmoid loss. During initialization, the substantial imbalance introduced by a multitude of negative samples prevails in the loss, prompting the need for substantial initial optimization steps to address this inherent bias [45]. Likewise, We observe the same phenomenon in vision pretraining and therefore utilize a learnable bias. The addition of learnable bias significantly enhances the performance of sigmoid loss, however, a fixed temperature is necessary to achieve the pinnacle performance of SigCLR.

We test SigCLR in widely used benchmarks such as CIFAR-10, and CIFAR100 [21], Tiny ImageNet [22] and ImageNet 100 [22] always outperforms SimCLR and have competitive results with other objectives. Thus, sigmoid contrastive learning stands as a simple yet powerful objective that later can be extended to other domains in which softmax is used.

2 Method

SigCLR learns representations by maximizing the similarity between distinctively augmented perspectives of a given data instance by applying the sigmoid contrastive loss within the latent space. We present the illustration of SigCLR in Figure 1. This alternative eliminates the need for calculating global normalization factors. The sigmoid-based loss operates independently on each image pair, transforming the learning task into a standard binary classification across all pair combinations in the dataset. Positive labels are assigned to pairs consisting of augmented views, while negative labels are assigned to all other pairs. The formulation of the loss is as follows:

$$-\frac{1}{|2\mathcal{B}|}\sum_{i=1}^{|2\mathcal{B}|}\sum_{j=1}^{|2\mathcal{B}|}\underbrace{k_{ij}\log\frac{1}{1+e^{z_{ij}(-t\mathbf{x}_i\cdot\mathbf{y}_j-b)}}}_{\mathcal{L}_{ij}}$$

where z_{ij} is the label for a given image inputs, which equals 1 if they are positive, and -1 otherwise. k_{ij} is 0 when x_i and y_j are the same view otherwise 1. Furthermore, We provide pseudocode in Algorithm A.3.

Table 1: Top-1 accuracies (%) under linear evaluation on different datasets. Results are reported from [11, 30]. We **bold** all top results that are statistically indistinguishable.

Method	CIFAR-10	CIFAR-100	Tiny-IN
	ResNet-18	ResNet-18	ResNet-18
SimCLR [7]	90.74	65.78	48.84
SimSiam [10]	91.40	66.04	-
BYOL [16]	92.58	70.46	51.00
W-MSE 2 [13]	91.55	66.10	49.22
MoCo-V2 [9]	**92.94**	69.89	-
Barlow [44]	92.10	70.90	-
VICReg [1]	92.07	68.54	-
SimCLR*	91.69	65.49	48.16
SigCLR	91.77	66.98	48.94

3 Experimental Setup

We perform our experiments on CIFAR-10, CIFAR-100 [21], Tiny-IN [22].

The experimental process consists of two successive phases: pretraining and linear evaluation. Initially, we employ the unsupervised pretraining of the encoder network using the proposed SigCLR method outlined in Section 2 on the training dataset. Once pretraining is completed, we proceed with linear evaluation—a standardized protocol used to assess the quality of the acquired representations, as established in prior research [7, 16].

We employ a variant of the ResNet-18 [19], similar to the configurations described in SimCLR [7]. The projector network is a three-layer MLP with ReLU activation functions for the hidden layers and linear activation functions for the output layer. The dimensions of the CIFAR-10, CIFAR-100, and Tiny-IN datasets are 1024-1024-128.

In the pretraining phase, we create two augmented versions of each input image by applying random cropping, resizing, and diverse operations including horizontal mirroring, color jittering, grayscale conversion, and Gaussian blurring. The augmentation parameters follow those of [30, 16]. In the training phase of linear evaluation, a single augmentation comprises random cropping, resizing, and a horizontal flip. During the test phase of linear evaluation, we use resizing and center crop augmentations, similar to the approach in [1, 44].

We employ 1000 epochs of pretraining on CIFAR-10, CIFAR-100, Tiny-IN. We use Lars [42] with cosine annealing and linear warmup, starting with a learning rate of 0.3 and linearly scaling with a starting batch size of 64. We incorporate a 10-epoch linear warmup with cosine annealing [25] and apply a weight decay of 0.000001.

Since we employ different data augmentations than the original SimCLR, we also reproduced SimCLR on CIFAR-10, CIFAR-100, and Tiny-IN. We report these results as SimCLR* in table 1 and also ablations in table 3.

Table 2: **Effect of bias and temperature** (CIFAR10 linear evaluation accuracy). In case of learnable bias, we initialize it to be -10. In all experiments batch size is 128.

	Temp. = 1	**Temp. = 2**	**Temp. = 5**	**Temp. = 10**
Fixed Temp. w/ Learnable Bias	90.76	91.25	91.53	89.80
Learnable Temp. w/ Learnable Bias	84.16	84.11	84.07	84.67
Fixed Temp. No Bias	87.17	84.22	27.15	17.37

Table 3: Results of SimCLR* across Datasets and Batch Sizes

	Batch Size				
Dataset	64	128	256	512	1024
CIFAR-10	90.56	91.69	92.23	92.42	92.26
CIFAR-100	62.85	65.49	66.67	67.26	66.49
Tiny-IN	46.08	48.16	49.92	49.16	49.94

Table 4: Results of SigCLR across Datasets and Batch Sizes

	Batch Size				
Dataset	64	128	256	512	1024
CIFAR-10	91.26	91.77	92.11	92.59	92.62
CIFAR-100	66.52	66.98	67.86	68.57	68.58
Tiny-IN	47.53	48.94	49.62	50.56	51.54

4 Results

Chen et al. [7] demonstrated that temperature-normalized logistic loss significantly underperforms compared to temperature-normalized softmax loss on ImageNet. To further investigate, we reevaluated this setup on CIFAR-10 and observed that models with lower temperatures still lag behind established SSL objectives, while higher temperatures lead to notably poor performance.

Additionally, we examined the most effective approach from SigLIP [45], which recommends both learnable temperature and bias. While the learnable bias helps maintain reasonable performance across different temperatures, the best performance achieved remains inferior to that of the standard logistic loss.

Consequently, we implement a setup using a learnable bias to maintain consistent performance across temperatures, paired with a fixed temperature to optimize results. With these two straightforward modifications, we find that the logistic loss achieves performance comparable to state-of-the-art SSL objectives.

We further extend the evaluation of SigCLR to CIFAR-100 and Tiny-IN, where it shows competitive performance, highlighting the effectiveness of the sigmoid contrastive loss.

Lastly, we assess both SimCLR* and SigCLR across varying batch sizes to investigate whether the large batch size issues seen in softmax-based contrastive learning are present. Our results indicate that SigCLR performs exceptionally well with smaller batch sizes, aligning with the findings of SigLIP.

5 Future Work and Limitations

Achieving results on ImageNet is crucial for establishing the credibility of SSL objectives. Therefore, the lack of such results considerably reduces the impact of our work. A key focus of future work is to extend our experiments to the final dataset, ImageNet-1k, as we believe this will provide a stronger foundation for our findings. However, the main limitation remains the significant computational resources required to train and evaluate this dataset.

6 Conclusion

We introduce SigCLR, a performant contrastive vision pretraining objective that employs sigmoid instead of the conventional softmax utilized in SimCLR. SigCLR demonstrates notable performance on established benchmarks, establishing itself as a strong contrastive objective. The straightforward objective of SigCLR eliminates the need for global normalization, a bottleneck for softmax, resulting in a highly efficient contrastive loss with enhanced performance.

References

[1] Adrien Bardes, Jean Ponce, and Yann LeCun. 2022. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning.

[2] Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. 2020. Are we done with ImageNet?

[3] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2019. Deep Clustering for Unsupervised Learning of Visual Features.

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2021. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments.

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging Properties in Self-Supervised Vision Transformers.

[6] Krishna Chaitanya, Ertunc Erdil, Neerav Karani, and Ender Konukoglu. 2020. Contrastive learning of global and local features for medical image segmentation with limited annotations.

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations.

[8] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. 2020. Big Self-Supervised Models are Strong Semi-Supervised Learners. *arXiv preprint arXiv:2006.10029*.

[9] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020. Improved Baselines with Momentum Contrastive Learning.

[10] Xinlei Chen and Kaiming He. 2020. Exploring Simple Siamese Representation Learning.

[11] Victor Guilherme Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. 2022. solo-learn: A Library of Self-supervised Methods for Visual Representation Learning. *Journal of Machine Learning Research*, 23(56):1–6.

[12] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. 2021. With a Little Help from My Friends: Nearest-Neighbor Contrastive Learning of Visual Representations.

[13] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. 2021. Whitening for Self-Supervised Representation Learning.

[14] Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. CERT: Contrastive Self-supervised Learning for Language Understanding.

[15] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2022. SimCSE: Simple Contrastive Learning of Sentence Embeddings.

[16] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. 2020. Bootstrap your own latent: A new approach to self-supervised Learning.

[17] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality Reduction by Learning an Invariant Mapping. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2:1735–1742.

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition.

[20] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization.

[21] Alex Krizhevsky. 2009. Learning Multiple Layers of Features from Tiny Images.

[22] Ya Le and Xuan S. Yang. 2015. Tiny ImageNet Visual Recognition Challenge.

[23] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation.

[24] Junnan Li, Pan Zhou, Caiming Xiong, and Steven C. H. Hoi. 2021. Prototypical Contrastive Learning of Unsupervised Representations.

[25] Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts.

[26] Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiwary, Paul Bennett, Jiawei Han, and Xia Song. 2021. COCO-LM: Correcting and Contrasting Text Sequences for Language Model Pretraining.

[27] Jovana Mitrovic, Brian McWilliams, Jacob Walker, Lars Buesing, and Charles Blundell. 2020. Representation Learning via Invariant Causal Mechanisms.

[28] OpenAI. 2023. GPT-4 Technical Report.

[29] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. 2023. DINOv2: Learning Robust Visual Features without Supervision.

[30] Serdar Ozsoy, Shadi Hamdan, Sercan Ö. Arik, Deniz Yuret, and Alper T. Erdogan. 2022. Self-Supervised Learning with an Information Maximization Criterion.

[31] Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive Learning for Many-to-many Multilingual Neural Machine Translation.

[32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision.

[33] Ana Sanchez-Fernandez, Elisabeth Rumetshofer, Sepp Hochreiter, and Günter Klambauer. 2023. CLOOME: contrastive learning unlocks bioimaging databases for queries with chemical structures. *bioRxiv*.

[34] Max Schwarzer, Johan Obando-Ceron, Aaron Courville, Marc Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. 2023. Bigger, Better, Faster: Human-level Atari with human-level efficiency.

[35] Yixuan Su, Tian Lan, Yan Wang, Dani Yogatama, Lingpeng Kong, and Nigel Collier. 2022. A Contrastive Framework for Neural Text Generation.

[36] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models.

[37] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation Learning with Contrastive Predictive Coding.

[38] Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. 2022. Molecular contrastive learning of representations via graph neural networks. *Nature Machine Intelligence*, 4(3):279–287.

[39] Zifeng Wang, Zhenbang Wu, Dinesh Agarwal, and Jimeng Sun. 2022. MedCLIP: Contrastive Learning from Unpaired Medical Images and Text.

[40] Ross Wightman, Hugo Touvron, and Hervé Jégou. 2021. ResNet strikes back: An improved training procedure in timm.

[41] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. 2022. Decoupled Contrastive Learning.

[42] Yang You, Igor Gitman, and Boris Ginsburg. 2017. Large Batch Training of Convolutional Networks.

[43] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2021. Graph Contrastive Learning with Augmentations.

[44] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. 2021. Barlow Twins: Self-Supervised Learning via Redundancy Reduction.

[45] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid Loss for Language Image Pre-Training.

[46] Jinghao Zhou, Li Dong, Zhe Gan, Lijuan Wang, and Furu Wei. 2022. Non-Contrastive Learning Meets Language-Image Pre-Training.

[47] Ömer Veysel Çağatan. 2024. UNSEE: Unsupervised Non-contrastive Sentence Embeddings.

A Appendix

A.1 Related Work

Sigmoid contrastive learning

The sigmoid is rather unpopular compared to InfoNCE [37] which greatly promotes the softmax-based contrastive learning however sigmoid has been used in unsupervised dimensionality reduction [17], supervised learning [40, 2].

SigLIP

While conventional methodologies rely on softmax normalization, demanding a comparison of each image against the entire dataset, SigLIP [45] takes a different approach by utilizing a sigmoid loss with learnable temperature and bias. This pairwise loss function directly operates on individual image-text pairs, eliminating the need for a comprehensive assessment of all pairwise similarities. This approach translates to significant enhancements in memory efficiency, allowing for the utilization of larger batch sizes and faster training, even when confronted with limited hardware resources. Notably, SigLIP simplifies implementation compared to softmax-based methods. Despite its straightforward design, models like SigLiT, built upon the foundation of SigLIP, have demonstrated state-of-the-art performance in tasks such as ImageNet zero-shot image classification, outperforming models trained with traditional contrastive learning. The combined benefits of accuracy, efficiency, and simplicity position SigLIP as a compelling alternative for pre-training models that adeptly handle visual-textual relationships

Contrastive image pretraining

SimCLR [7] stands out due to its simplicity and adaptability. It employs a single encoder to extract representations from two augmented versions of an image. The subsequent comparison of these representations involves a temperature-scaled normalized cross-entropy loss. This process encourages the identification of features distinguishing positive pairs from negative ones with the temperature parameter adjusting the difficulty and fostering fine-grained feature discrimination.

On the other hand, MoCo [18] takes a different approach by using a two-encoder system: a query encoder for generating representations and a momentum encoder that gradually catches up to the query. Positive pairs are formed during training as each image passes through both encoders. MoCo uniquely leverages a queue of past representations, enabling the query encoder to learn from a diverse

set of examples beyond the current batch. This queue acts as a historical memory bank, enhancing the overall training process.

NNCLR [12] acts as a bridge between SimCLR and MoCo, utilizing a single encoder like SimCLR but with a twist in forming positive pairs. Instead of relying on two augmented views of the same image, NNCLR incorporates a memory bank similar to MoCo's queue. However, it identifies positive pairs by locating the k nearest neighbors of the current query representation in the memory bank. This neighbor-based approach focuses on representations sharing similar features, further refining the learned features through a dedicated nearest-neighbor contrastive loss.

Non-Contrastive image pretraining Recent progress in visual self-supervised learning extends beyond the conventional contrastive paradigm, exploring innovative approaches to diminish reliance on negative samples. These methods are purposefully crafted to enhance the quality of augmented representations, operating independently of negative samples and forming a subset known as non-contrastive objectives. To tackle challenges like model collapse, various strategies have been introduced, including the incorporation of asymmetric architectures [16, 10], utilization of feature decorrelation techniques [1, 44, 13, 30], and the integration of clustering methods [4, 3].

A.2 Efficient Chunked Sigmoid Implementation

The anticipated strength of the sigmoid loss function lies predominantly in distributed training scenarios. While in single-device training, the softmax loss still necessitates two passes of data, it appears to have minimal impact. In contrast, the softmax loss tends to exhibit slightly faster performance, attributed to the inherent masking operations of the sigmoid loss. Although the observed difference is marginal, it becomes more apparent and advantageous when scaling up to distributed training environments. The distinctive characteristics of the sigmoid loss make it particularly well-suited for handling distributed training workloads, where efficiency and scalability are paramount considerations. Thus, we also utilize the efficient chunked sigmoid implementation from SigLIP [45].

Contrastive training often utilizes data parallelism, which involves dividing data across D devices. However, computing the loss in this setup requires collecting all embeddings [45], which entails costly all-gathers. Moreover, it mandates the creation of a memory-intensive $|\mathcal{B}| \times |\mathcal{B}|$ matrix for pairwise similarities. In contrast, the sigmoid loss is better suited for a memory-efficient, fast, and numerically stable implementation, effectively addressing both challenges. By expressing the per-device batch size as $b = \frac{|\mathcal{B}|}{D}$, the loss can be redefined as:

$$-\frac{1}{|\mathcal{B}|} \underbrace{\sum_{d_i=1}^{D}}_{\textbf{A: } \forall \text{ device } d_i} \overbrace{\sum_{d_j=1}^{D}}^{\textbf{B: swap negs across devices}} \overbrace{\underbrace{\sum_{i=bd_i}^{b(d_i+1)}}_{\text{all local positives}} \underbrace{\sum_{j=bd_j}^{b(d_j+1)}}_{\text{negs from next device}} \mathcal{L}_{ij}}^{\textbf{C: per device loss}}$$

This is particularly straightforward for the sigmoid loss since each pair serves as an independent term in the loss function. In essence, we start by calculating the loss component corresponding to the positive pairs and subsequently address $b - 1$ negative pairs in the computation.

Afterward, we permute representations among devices, enabling each device to incorporate negatives from its adjacent device in the subsequent iteration of the summation (**B**). The loss is subsequently computed to this chunk (sum **C**). This process is carried out independently on each device, ensuring that each device calculates the loss based on its local batch b.

Losses can be efficiently computed by aggregating them across all devices (designated as **A**). The individual collective permutes for the sum (**B**) exhibit fast performance, often surpassing the speed of two all-gathers between D devices. Simultaneously, the memory requirement at any given moment is diminished from $|\mathcal{B}|^2$ to b^2 (for the sum **C**). Typically, b remains constant, as the scaling of $|\mathcal{B}|$ is accomplished by increasing the number of accelerators. However, the vanilla loss computation faces rapid scaling bottlenecks due to its quadratic dependency on the batch size [45].

A.3 Pseudocode

Algorithm 1 Jax-like sigmoid loss pseudocode.

```
# img_emb: embedding of augmented views [2n, dim]
# t, b     : fixed temperature and learnable bias
# n        : mini-batch size
# cos_sims : cos_sim of all pairs [2n,2n]
# sim_mask : positive-negative pairs mask
# loss_mask: mask for loss of the same view

diag_range = arange(2n)
shift_range = roll(diag_range,n)

sim_mask = -ones(2n,2n)
loss_mask = ones(2n,2n)
sim_mask[diag_range,shift_range] = 1
loss_mask[diag_range,diag_range] = 0

emb_1, emb2 = img_emb[:,None,:],img_emb[None,:,:]
cos_sims = t * cos_sim(emb_1,emb2) + b

log_lik = -log_sigmoid(cos_sims * sim_mask)
loss = mean(log_lik * loss_mask)
```