Self-Supervised Contrastive Learning for Vision Representations

Amisha Himanshu Somaiya
Dept. of Electrical and Computer Engineering
University of Washington
Seattle, WA 98195
amishahs@uw.edu

Megha Chandra Nandyala
Dept. of Electrical and Computer Engineering
University of Washington
Seattle, WA 98195
nvmcr@uw.edu

Abstract

This project addresses the challenge of unavailability of annotations in real world vision datasets by generating representations by extracting useful features from raw unlabeled data. The project self-implements simCLR a popular framework for self-supervised contrastive learning for generating representations. This implementation is then evaluated on downstream tasks of Linear Probing and Fine-Tuning. simCLR is computationally expensive and the original paper implementation uses a default batch size of 4092 and 32 TPUs for good performance. Another issue with simCLR is the positive-negative coupling effect in the InfoNCE loss, due to which the model performance degrades at sub-optimal hyperparameters like small batch sizes. This project addresses these issues by applying Decoupled Contrastive Learning (DCL) loss instead of simCLR loss. The final project implementation of simCLR with DCL loss with Resnet50 backbone at 100 epochs achieves a top1 accuracy of 84.84% at batch size of 32 for linear evaluation on CIFAR10 dataset. This outperforms the top1 accuracy of 81.66% at the same batch size of 32 with simCLR loss and has comparable performance to previously obtained top1 accuracy of 85.08% at higher batch size of 128. Thus, the final project implementation of simCLR with DCL efficiently generates vision representations at lower batch sizes with computational savings and good performance metrics that can be used for several downstream vision tasks.

1 Introduction

Machine Learning algorithms for computer vision tasks have applications in real world scenarios. However, most of the vision real world datasets are unannotated. The motivation for this project stems from our recent work in ENGINE Capstone 2023 where we had extremely limited and unannotated real data. We combatted this challenge by zero-shot synthetic data generation with semantic labels using UnReal Engine 5.1.1 and followed this module by Unsupervised Domain Adaptation with training on source domain (synthetic data) and testing on target domain (real industry data). However, this approach has a learning curve for data generation software (Unity, UnReal Engine, Blender etc.) and hence is more suited when data itself is unavailable or scarce. Since most real-world datasets have massive raw data i.e. image/video data but are unannotated, we are motivated to seek a generic approach. In this project we combat this challenge by self-implementing simCLR : a self-supervised constrastive learning framework for learning vision representations. We further complement our implementation with Decoupled Contrastive Learning loss so the final implementation of simCLR with DCL efficiently generates vision representations at lower batch sizes with computational savings and good performance metrics that can be used for several downstream vision tasks.

2 Related Work

The basic approach to tackle unavailability of labels is to hand-annotate the dataset and perform supervised learning. Even though such supervised learning will lead to greater performance metrics due to ease of learning from ground truths, hand-annotating is labor-intensive and time-consuming and hence not a viable approach. Thus, learning effective visual representations without human supervision is needed [1]. Representation learning aims to automatically extract useful information from the raw data which can be used for downstream tasks. It facilitates feature learning with its competence in exploiting massive raw data without any annotated supervision [2]. Most mainstream approaches for representation learning fall into one of two classes: generative or discriminative. Generative approaches learn to generate or model pixels in the input space [3]. Discriminative approaches learn representations using objective functions similar to those used for supervised learning, but train networks to perform pretext tasks where both the inputs and labels are derived from an unlabeled dataset. Many such approaches have relied on heuristics to design pretext tasks [1].

In the early stages of SSL, representation learning focused on exploiting pretext tasks, which are addressed by generating pseudo-labels to the unlabeled data through different transformations, such as solving jigsaw puzzles [4], colorization [5] and rotation prediction [6]. Though these approaches succeed in computer vision, there is a large gap between these methods and supervised learning. Contrastive learning for self-supervised pre-training significantly bridge the gap between the SSL methods and supervised learning [2]. Contrastive SSL methods try to pull different views of the same instance close and push different instances far apart in the representation space. Despite the evident progress of the state-of-the-art contrastive SSL methods, there are several challenges. The SOTA models, e.g., [7] may require specific structures such as the momentum encoder and large memory queues, which may complicate the underlying representation learning [1].

3 Algorithm Design

3.1 SimCLR: A Simple Framework for Contrastive Learning of Visual Representations

SimCLR [1] is a self-supervised contrastive learning framework that generates vision representations from raw unlabeled data. These representations can then be used for several downstream tasks. SimCLR outperforms previous methods and does not require specialized architectures or a memory bank. The major components of SimCLR framework are (1) Data Augmentation : plays a crucial role in defining effective predictive tasks (2) Projection Head : a learnable nonlinear transformation between the representation and the contrastive loss which substantially improves the quality of the learned representations (3) Contrastive Loss : simultaneously maximize agreement between differently transformed views of the same image, so the Representations of corresponding views to "attract" each other and minimize agreement between transformed views of different images, so Representations of non-corresponding views to "repel" each other.

SimCLR algorithm (shown in figure 1) randomly samples a minibatch of N examples and defines the contrastive prediction task on pairs of augmented examples derived from the minibatch, resulting in 2N data points. It does not sample negative examples explicitly. Instead, given a positive pair, it treats the other 2(N − 1) augmented examples within a minibatch as negative examples. Let $\mathrm{sim}(u, v) = \frac{u^T v}{||u|| \cdot ||v||}$ denote the dot product between l_2 normalized u and v (i.e. cosine similarity). Then the loss function for a positive pair of examples (i, j) is defined as

$$l_{i,j} = -\log\left(\frac{\exp(\mathrm{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\mathrm{sim}(z_i, z_k)/\tau)}\right) \quad (1)$$

where $1_{[k \neq i]} \in \{0,1\}$ across all positive pairs, both (i, j) and (j, i), in a mini-batch, termed as NT-Xent (the normalized temperature-scaled cross entropy loss).

input: batch size N, temperature τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\tau \|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i, j) = -\log \frac{\exp(s_{i,j})}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k})}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network f

Figure 1: SIMCLR's Algorithm

3.2 Limitations

3.2.1 Computational Resources

Training SimCLR requires significant computational resources, including high-performance GPUs or TPUs and large memory capacity. The large-scale training process, involving training on large batches (~4k to 9k), large training epochs. and extensive data augmentations, can be computationally intensive and time-consuming.

3.2.2 Hyperparameter Sensitivity

As with many deep learning techniques, SimCLR's performance is influenced by various hyperparameters, such as the batch size, learning rate, temperature parameter, and the choice of augmentation techniques. Finding the optimal combination of hyperparameters can require extensive experimentation and tuning.

3.2.3 NPC Effect

The issue of the Negative Positive Coupling (NPC) effect in the loss arises from the way negative samples are chosen during training. When we take a batch of images, we select the positive pairs and all the remaining samples are treated as negative. When we have a large batch size, there wouldn't be any issue as there are enough negative samples for the model to learn underlying representations. But with a lower batch size, the negative samples chosen for a positive pair are too easy to distinguish from the positive sample, leading to an overly optimistic loss landscape.

3.3 Decoupled Contrastive Learning Loss

Training SimCLR requires significant computational resources, including high-performance GPUs Decoupled Contrastive Learning (DCL) is a modification of the contrastive learning framework, specifically designed to address some of the limitations and challenges associated with the positive-negative coupling effect in the InfoNCE loss. It aims to improve the learning of

discriminative representations by decoupling the positive and negative samples during the training process.

DCL introduces a decoupling mechanism by considering positive and negative pairs separately in the loss function. Instead of using a single temperature parameter to control the contrastive loss, DCL utilizes two separate temperature parameters: one for the positive pairs and another for the negative pairs. This decoupling allows for more flexibility in optimizing the positive and negative sample representations independently.

The decoupled contrastive loss used in DCL can be expressed as follows:

$$L_{\mathrm{DC}} = \sum_{k\in\{1,2\}} \sum_{i\in[1,N]} L_{DC,i}^{\{k\}}, \text{ where } L_{(k)}^{\mathrm{DC},i} \text{ is:}$$

$$L_{DC,i}^{\{k\}} = \left(z_i^{(1)}, z_i^{(2)}\right)/\tau + \log U_{i,k} \quad (2)$$

4 Design Implementation

4.1 Data Loader

Figure 2 shows the used data loader with paper [1] implementation in blue color.

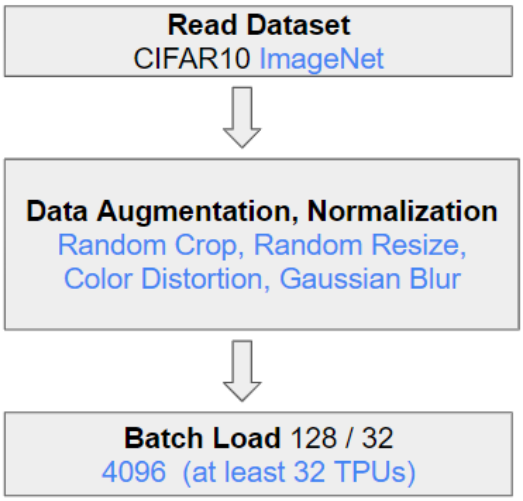

Figure 2: Data Loader

4.1.1 Dataset and Framework

The original TensorFlow implementation uses ImageNet dataset with 1.2 million high-resolution RGB images in 1000 classes. We use PyTorch framework on CIFAR10 dataset with 60,000 32x32 RGB images in 10 classes.

4.1.2 Data Augmentation Module

We augmented the training data with Random Flip, Random Crop & Resize, Color Distortion and with and without Gaussian Blur in our experiments. Each training of 100 epochs took us 12 hours on a single GPU. Since our goal was to better the SimCLR performance metrics at lower batch sizes than existing values, we wanted to train our network with the best. Hence, as per the SimCLR and DCL papers, we retained random resize, color distortion and gaussian blur for our SimCLR experiments and removed Gaussian Blur for DCL experiments.

4.1.3 Batch Size and Computational Resources

The original implementation does not train the model with a memory bank and hence uses a large batch size with a default of 4096 samples. This gives 8192 negative examples per positive pair from both augmentation views. Training with such large batch sizes needs at least 32 TPUs. Thus,

for computational reasons, we use a batch size of 128 and 32 on a single NVIDIA GeForce RTX 2080 Ti/PCIe/SSE2 GPU.

4.2 Training Loop Forward Pass

4.2.1 Encoder

The encoder *f(.)* extracts representation vectors from augmented data examples. The original implementation has several choices for network architecture but uses wide network Resnet50 4x with ~ 375 M total number of parameters for best metrics. We train on Resnet50 1x and Resnet18 1x backbones with ~ 23.56M parameters. The output after pooling is $h_i = f(\tilde{x}_i) = \text{ResNet}(\tilde{x}_i)$ where $h_i \in R^d$

4.2.2 Projection Head

The projection head *g(.)* enhances the quality of representations and maps representations to the space where contrastive loss is applied. It is an MLP with 1 hidden layer to obtain $z_i = g(h_i) = W^{(2)}\sigma(W^{(1)}h_i)$ where σ is a RELU non-linearity.

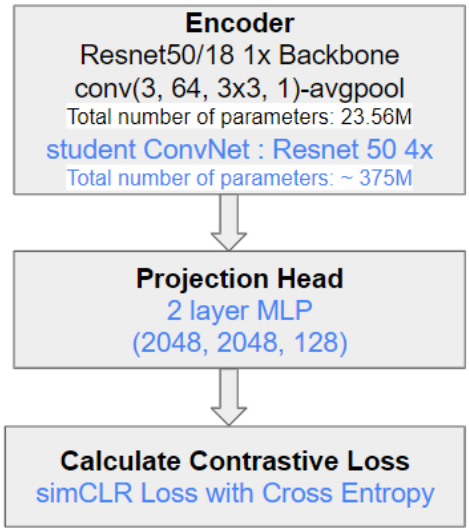

Figure 3: Forward Pass

4.2.3 Contrastive Loss Function

Given a set $\{x_k\}$ including a positive pair for examples x_i and x_j, the contrastive prediction task aims to identify x_j in $\{x_k\}_{k\neq i}$ for a given x_i.

4.3 Training Loop Backward Pass

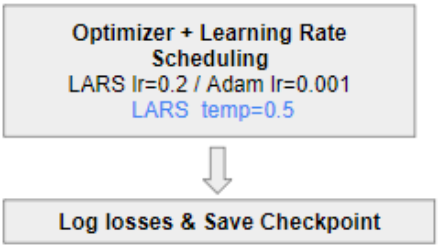

Figure 4: Backward Pass

4.3.1 Optimizer and Learning Rate Scheduling

Optimizer computes gradients of model parameters and updates these during backward pass to reduce the contrastive loss. The original implementation uses LARS optimizer to stabilize the training at large batch sizes since training with large batch size becomes unstable when using standard SGD/Momentum with linear learning rate scaling. We use LARS with a learning rate of 0.2 for SimCLR experiments and switch to Adam with learning rate of 0.001 for DCL experiments at lower batch sizes.

The learning rate schedule used is linear warmup for the first 10 epochs and decay the learning rate with the cosine decay schedule without restarts. Training loss is logged at every epoch and model checkpoints are saved at 50 and 100 epochs.

4.4 Downstream Tasks

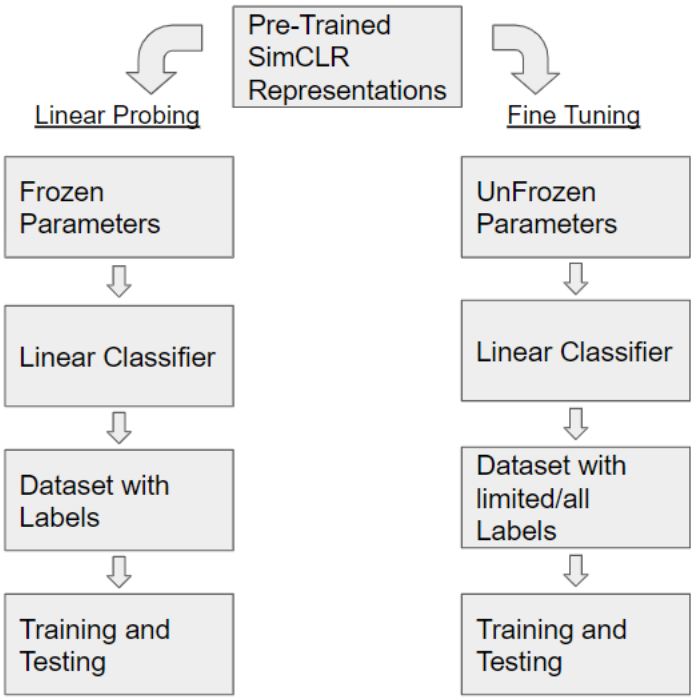

Figure 5: Downstream Block

The learnt representations from the training loop are frozen and can be used for several downstream tasks. For the CIFAR10 dataset, we perform the downstream tasks of Linear Probing and Fine-tuning. In linear probing, the trained model checkpoint from training loop is used to train the linear classifier and evaluated on the test data. In fine tuning, the model is trained again on just 1% labeled data.

5 Experimental Results

Table 1 presents a comparison of the results obtained using different batch sizes and model layers. Our findings indicate that utilizing a larger batch size and a more substantial model leads to superior linear evaluation outcomes. Furthermore, we observed even greater performance enhancements by fine-tuning the model using only 1% labeled data. Figure 6 shows the training and validation curves plotted during training of the SimCLR model with ResNet-50 backbone using 128 batch size along with TSNE features after epoch 10 and 100.

Table 1: Comparison of model size and batch size with Contrastive Loss

Model	Batch Size	Evaluation	Loss	Test Accuracy (%)
ResNet-50	128	Linear Probing	0.458	85.08
ResNet-50	32	Linear Probing	1.8	81.66
ResNet-18	128	Linear Probing	0.599	84.93
ResNet-18	32	Linear Probing	1.73	80.97
ResNet-50	128	Fine Tuning	0.458	86.7

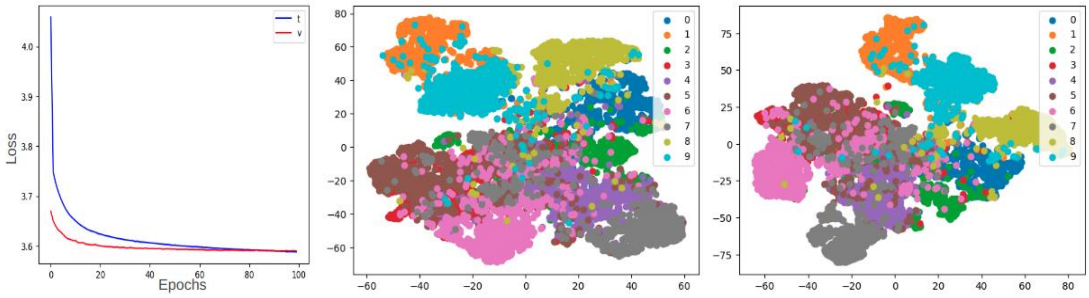

Figure 6: Plots of SimCLR model

As discussed in limitations, there is a need for a different approach. Table 2 compares the results achieved through linear probing using different loss functions. The findings indicate that performance of DCL is similar to contrastive loss when employing larger batch sizes, while demonstrating enhanced performance at lower batch sizes.

Table 2: Comparison of model performance with DCL and Contrastive Loss

Model	Batch Size	Loss Function	Loss	Test Accuracy (%)
ResNet-50	128	DCL	0.42	85.42
ResNet-50	**32**	**DCL**	**0.62**	**84.4**
ResNet-50	128	Contrastive	0.458	85.08
ResNet-50	32	Contrastive	1.8	81.66

6 Conclusion

Self-supervised Contrastive Learning by SimCLR pre-training from raw unlabeled data creates vision representations that are useful for several downstream tasks. However, SimCLR suffers from positive-negative coupling effect and hence suffers performance degradation at sub-optimal hyper-parameters such as lower batch size. The final implementation by combining SimCLR with DCL addresses this issue and achieves matching accuracy scores even at lower batch sizes making this pipeline low cost, robust & less sensitive to suboptimal hyperparameters.

7 Future Work

We would like to extend our project to multi-modality applications. In lines of ‘SLIP: Self-supervision meets Language-Image Pre-training’ [8] by Facebook-Research, we wish to combine

our SimCLR + DCL implementation to CLIP and achieve improved baselines for 3 vision-text intersection tasks of Image Captioning, Visual Question Answering and Image Retrieval.

Acknowledgments

We thank Prof. Linda Shapiro and the teaching staff especially TA Mehmet Saygin for their guidance and support.

References

[1] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," arXiv.org, https://arxiv.org/abs/2002.05709.

[2] C.-H. Yeh et al., "Decoupled contrastive learning," arXiv.org, https://arxiv.org/abs/2110.06848.

[3] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., WardeFarley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014

[4] Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: *European Conference on Computer Vision* (ECCV) (2016)

[5] Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: *European Conference on Computer Vision* (ECCV) (2016)

[6] Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: *International Conference on Learning Representations* (ICLR) (2018)

[7] He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (CVPR) (2020)

[8] N. Mu, A. Kirillov, D. Wagner, and S. Xie, "Slip: Self-supervision meets language-image pre-training," arXiv.org, https://arxiv.org/abs/2112.12750.