An evaluation of self-supervised learning frameworks – SimCLR and RotNet

Bernie Chen*
Duke University
Bernie.Chen@duke.edu

Ethan Hsu*
Duke University
Ethan.Hsu@duke.edu

Michael Jang*
Duke University
Jungkyu.Jang@duke.edu

Abstract

This paper introduces and explores two popular self-supervised learning frameworks. We demonstrate the effectiveness of these learning methods in different settings. In particular, we show that (1) batch size and number of training steps pose positive correlation with the representation learnt, (2) the backbone model serves as an effective initialization for various fine-tuning task, (3) self-supervised learning relies on large number of parameters to achieve good representation and can be sensitive to adversarial attacks.

1 Introduction

Modern machine learning is usually empowered by performing supervised learning on large number of labeled data. However, a high quality and quantity of labeled data can be expensive and erroneous. Self-supervised learning (SSL) method enabled the potential for models to learn data representation without the needs of labels. Unlike unsupervised learning, which concentrates on learning specific data patterns, clustering, and detecting anomaly, SSL aims to utilize and obtain "labels" from the data itself.

In this paper, we investigate two common approaches on representation learning – SimCLR [1] and RotNet [2]. Section 2 discusses the related background knowledge, section 3 describes the detailed experimental setup, and section 4 demonstrates the results we acquire in this paper. Finally, a thorough discussion of the implication and impact of our results can be found in section 5. The complete implementation can be found at https://github.com/EtHsu0/Duke_ECE661_Final-Project-An_Evaluation_of_Self-Supervised_Learning_Method-SimCLR_and_RotNet.

2 Related Works

2.1 Representation Learning via Contrastive Loss – SimCLR

There is a great family of self-supervised learning frameworks that is based on the principle of encouraging similarity between transformed versions of input, which is often referred to as deep metric learning [3] or contrastive learning [4]. A Simple Framework for Contrastive Learning of Visual Representations (SimCLR) [1] adopted the contrastive learning loss as defined in equation 1. ℓ defines the loss of each positive pair example using cosine similarity and cross entropy. The total loss $\mathcal{L}$ sums up the contrastive loss of every positive pair.

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z_i}, \boldsymbol{z_j})/\tau}{\sum_{k=1}^{2N} \mathbb{1}_{k \neq i} \exp(\text{sim}(\boldsymbol{z_i}, \boldsymbol{z_k})/\tau)}, \quad \mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)] \tag{1}$$

In SimCLR, a positive pair is generated by taking an image and performing two random transformations on the same image. As shown on the left in Figure 1, the two transformations are created and passed into an encoder, denoted as $f(\cdot)$. This encoder outputs a learned representation $\boldsymbol{h_i}$, which is passed into a projector $g(\cdot)$ to produce an output $\boldsymbol{z_i}$ for contrastive loss. The goal for SimCLR is to use unlabeled dataset and image transformation for the encoder to learn a high quality representation.

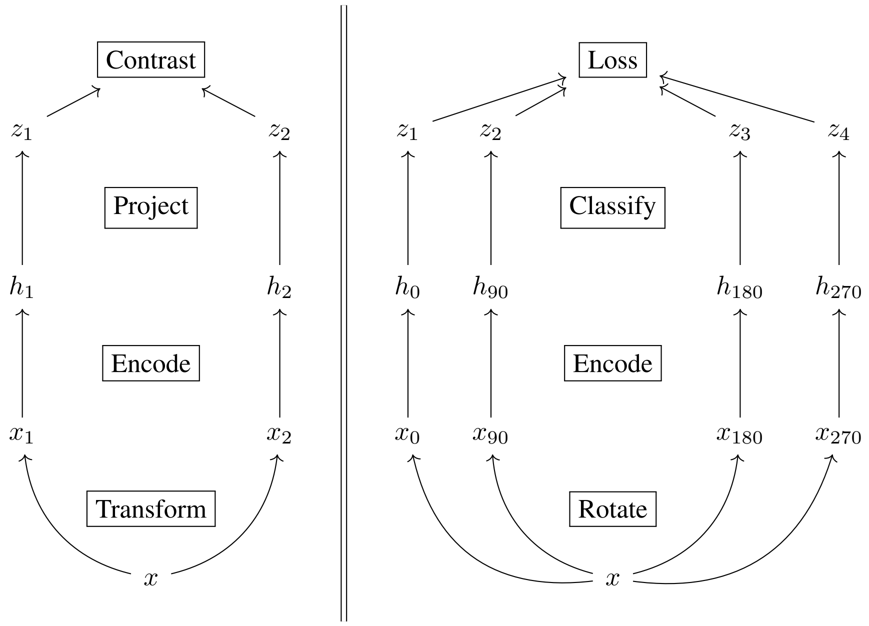


Figure 1: SimCLR (Left) and RotNet (Right) Framework Diagram

2.2 Representation Learning via Predicting Transformation – RotNet

RotNet [2] is another popular self-supervised learning approach, which uses the property of the data to generate labels and train the model with the standard empirical risk minimization (ERM) method. Specifically, the right of Figure 1 demonstrates the procedure, where one image is transformed into four rotation views – 0, 90, 180, and 270 degrees. All images are then passed into the model to generate a representation just like SimCLR. A classifier then turns representations into predictions, and the model is optimized by maximizing the probability of predicting these four rotation labels correctly.

2.3 Linear Evaluation

Unlike supervised learning, it is not immediately obvious on how to evaluate model performances. When labeled data is available, linear evaluation [5] is the most popular protocol to evalute the model. To perform linear evaluation, a linear classifier is appended on a pretrained encoder. Then, the linear classifier is trained using the available labeled data with the encoder weight frozen to fix feature representation. In short, we take the representation $\boldsymbol{h}$ and perform linear-probing evaluation with labeled dataset. This method has been most popular because it achieves high accuracy and relies heavily on the quality of representation learnt. It is also very computationally efficient, only requiring few epochs.

2.4 Semi-Supervised Learning

Self-supervised models are only capable of learning representations that can be useful for various downstream tasks and applications. Semi-supervised learning also seeks to take advantage of unlabeled data but uses labeled data to boost or assist self-supervised methods [6]. For image classification, semi-supervised learning refers to using limited amounts of labeled data to help strengthen representations learned from self-supervised models. These strengthened representations can be compared to those from fully supervised baselines trained on limited labeled data [7].

2.5 Adversarial Attacks

Adversarial examples are inputs designed to cause incorrect classification on machine learning models. Specifically, a whitebox attack is a case of an adversarial attack when the entire model architecture and weights are known prior to generating adversarial examples. Two popular methods for such an attack are the fast gradient sign method (FGSM) attack [8] and the projected gradient descent (PGD) attack [9]. For FGSM, gradient ascent is used to maximize the loss to generate an adversarial example $X^{\text{adv}} = X + \epsilon \text{sign}(\nabla_X J(X, y_{true}))$. PGD is a multi-step variant of the FGSM attack with the choice of random start near the data sample $X_0^{\text{adv}} \sim U(X - \epsilon, X + \epsilon)$. This gives $X_{N+1}^{\text{adv}} = \text{clip}_{x,\epsilon}(X_N^{\text{adv}} + \alpha \text{sign}(\nabla_X J(X_N^{\text{adv}}, y_{true})))$. It is known to be the strongest attack utilizing the local first order information.

2.6 Transfer Learning

For our purposes, there are two main types of transfer learning. The first takes a pretrained model as a fixed feature extractor and simply maps extracted features to help with different tasks. The second involves fine-tuning the pretrained weights and follows the intuition of fine-tuning on pretrained weights rather than training on random weights. In the context of self-supervised learning, the first method corresponds to the linear evaluation detailed in section 2.3. The second method is often used for evaluating models on transferring to different datasets [10]. For self-supervised methods, transfer learning protocols and metrics have been established for comparisons [5].

3 Methodology

3.1 Self-supervised Learning Setup

For SimCLR, we use ResNet-50 [11] as the encoder, a multi-layer perceptron (MLP) with rectified linear unit (ReLU) as the projector. We defined the contrastive loss as described in equation 1 and use Adam optimizer with learning rate of 10^{-3}, weight decay of 10^{-6}, and projection output dimension 2048.

For RotNet, the original paper uses a network in network (NIN) [12] for CIFAR-10 experiment, and uses AlexNet [13] for ImageNet experiments. In our paper, we also use ResNet-50 with the intention of directly comparing the representations learned using the two different methods. We use standard cross-entropy loss and stochastic gradient descent with Nesterov momentum (NSGD) with learning rates of 0.01, momentum of 0.9, and weight decay of 5×10^{-4}.

For both methods, we pretrain the model on the CIFAR10 dataset for 5000 steps, each step represents one model weights update. CIFAR10 [14] includes 50,000 training images and 10,000 test images with 10 classes. We perform the same pretrain routine on different batch sizes for both models as shown in 4.1.

3.2 Semi-Supervised Learning Evaluation Setup

A fully-connected layer is appended to the encoder pretrained with 1000 steps and batch size of 512. This model is fine-tuned on fractions of labeled data. We trained on 1%, 10%, and 50% of CIFAR10 training set with batch size of 64. We used cross-entropy loss, NSGD optimizer with momentum of 0.9, and a learning rate of 0.0125 ($0.05 \times \text{BatchSize}/256$) with no weight decay or regularization. Random cropping and horizontal flip were applied to the training images. As suggested in [1], we ran 60 epochs for 1% labeled data and 30 epochs for 10%. For 50% labeled data, we run 20 epochs.

In addition, we trained a supervised baseline using ResNet-50 on same parameters but with 500 epochs. Results and observations are detailed in section 4.2.

3.3 Adversarial Attacks Setup

To evaluate and compare the robustness of SimCLR pretrained and finetuned semi-supervised model, we perform a whitebox PGD and FGSM attacks on the supervised baseline and semi-supervised model trained with 1%, 10%, and 50% labeled data. For the attack strength, 10 linearly spaced ϵ from

0 to 0.275 were used. 10 iterations of gradient ascent with $\alpha = 1.85$ was used for PGD to generate each adversarial data.

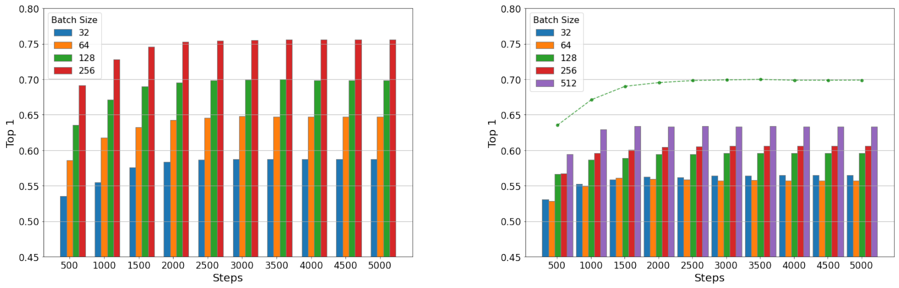
3.4 Transfer Learning Evaluation Setup

For transfer learning, we use STL10 [15], which is a dataset suitable for unsupervised learning tasks. There are 10 classes, composed of 100,000 unlabeled images. For supervised training, it has 500 labeled training images and 8000 test images. We perform transfer learning on the supervised baseline and SimCLR models.

To compare the effect of dataset size and step numbers on transfer learning, we pretrain models on CIFAR10 and STL10 with batch size of 128, saving the models at various training steps. In addition, we also take the SimCLR model that was pretrained on various batch sizes using CIFAR10. Then, each model is evaluated using linear protocol on the counterpart dataset. The accuracy of each model is reported in section 4.4.

4 Results

4.1 Self-Supervised Linear Comparison Results



(a) SimCLR

(b) RotNet with SimCLR baseline (green line).

Figure 2: Linear Evaluation Results

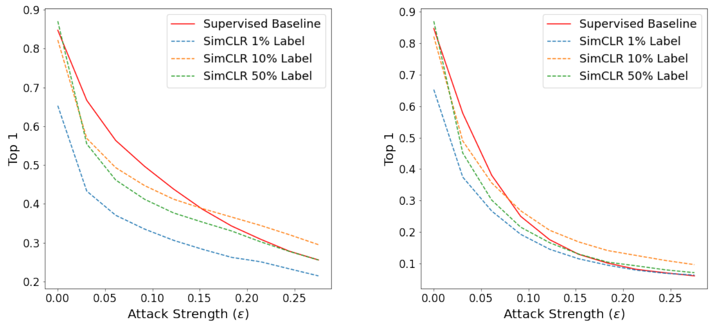
Figure 2 shows the result of performing linear evaluation on various pretrained models. Both SimCLR and RotNet benefit from larger batch size and steps. Also, SimCLR outperforms RotNet under the same conditions, demonstrating the effectiveness of contrastive learning over standard ERM with generated labels.

4.2 Semi-Supervised Learning Results

Metric	Supervised ResNet-50			Fine-tuned RotNet			Fine-tuned SimCLR		
	1%	10%	50%	1%	10%	50%	1%	10%	50%
Top-1 Accuracy	0.141	0.373	0.475	0.515	0.560	0.577	0.653	0.822	**0.870**
Top-2 Accuracy	0.260	0.539	0.682	0.721	0.760	0.771	0.822	0.929	**0.953**
AUC	0.634	0.787	0.873	0.881	0.899	0.909	0.935	0.981	**0.990**

Table 1: Comparison of Semi-Supervised Learning with SSL vs Supervised Baseline

Table 1 compares fine-tuned SSL methods to the supervised baseline. As observed in the linear evaluation results, SimCLR pretrained model outperforms RotNet for the downstream semi-supervised learning task. Both SSL methods surpass the supervised baseline with significantly fewer epochs as mentioned in 3.2.



(a) FGSM

(b) PGD

Figure 3: Whitebox Attack Results

4.3 Adversarial Attack Results

Figure 3 shows the results of conducting PGD and FGSM attack on the fine-tuned semi-supervised models and the supervised baseline. Both results show that SimCLR pretrained and finetuned model are not robust to even a small rate of attack strength, especially highlighted in the case of FGSM. PGD results also hints the adversarial susceptibility of the SimCLR framework, but is less significant compared to FGSM as PGD is a much stronger attack that effectively targets both the semi-supervised and supervised models.

4.4 Transfer Learning Evaluation Results

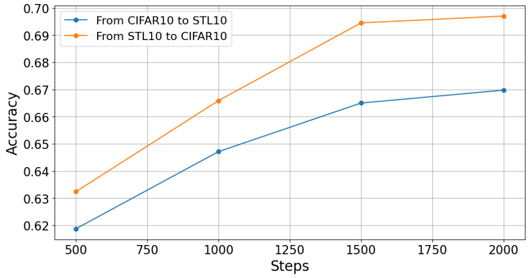


Figure 4: CIFAR10 SimCLR encoder transferred to STL10 data versus STL10 SimCLR encoder transferred to CIFAR10 data.
NOTE: Supervised Baseline ResNet-50 CIFAR10 transferred to STL10 accuracy at 2000 steps: **0.4713**

Table 2: Transfer Learning Tabulated Results

(a) Performance Comparison of Transfer Learning Models

Transfer Learning Model	Top-1	Top-2	Top-3	AUC
CIFAR10 to STL10	0.4713	0.6838	0.7913	0.8783
CIFAR10 to STL10	0.6698	0.8406	0.9176	0.9482
STL10 to CIFAR10	**0.6970**	**0.8612**	**0.9270**	**0.9542**

(b) Batch Size vs Top-1 (CIFAR10 to STL10)

Batch	Top-1
Batch 32	0.5716
Batch 64	0.6304
Batch 128	0.6740
Batch 256	**0.7105**

Both CIFAR10 and STL10 pretrained models outperform the supervised baseline in terms of transfer learning across all metrics shown in Table 2a. Similar to the results in section 4.1, increasing the number of training steps and batch size for the pretrain routine increases test accuracy at final evaluation as shown in Figure 4. The increase in accuracy also plateaus at around 2000 steps which matches the results in Figure 2. We also observe that transfer learning also benefits from larger batch size. Table 2b shows that increasing encoder training batch size increases accuracy for the CIFAR10 transferred to STL10. This trend was also observed for self-supervised evaluation. When comparing the performance of transfer learning with different datasets in Figure 4 and Table 2a, we see that the model pretrained on STL10 dataset performs better than CIFAR10 consistently. This indicates a positive correlation of dataset size and the quality of learned representation.

5 Discussion and Conclusion

In section 4.2 and 4.4, we showed SSL pretrained models are good backbones for downstream classification tasks. With few epochs of finetuning, these learned representations achieve high accuracies on different tasks to outperform supervised methods. This means SSL methods can serve as a great way for weights initialization even in traditional supervised learning. This allows fast convergence during model training, making it possible to perform exhaustive hyper-parameter tuning or other task. However, these benefits rely on the assumption that the SSL method learns meaningful representation and semantics on the dataset, which can be difficult outside of the dataset we used.

Between the two SSL method we examined, we see SimCLR outperforms RotNet in both sections 4.1 and 4.2. Using the same architecture, SimCLR outperforms RotNet likely because it uses a diverse set of image transformations to extract features with a designated contrastive learning objective. Yet, RotNet is much more computationally efficient. We conclude that SimCLR learns better representations of images compared to RotNet using the same ResNet-50 encoder architecture.

For the semi-supervised evaluation, we hypothesize that if the clusters generated from the self-supervised SimCLR encoder in the latent-space represent relative distances between images well, having labeled data to fine-tune the model for a classification task will be more effective than mapping limited features extracted from rotations to image classes. Since SimCLR only treats transformations of the same image as positive pairs, it also treats different images of the same class as negative pairs. We hypothesize that fine-tuning SimCLR with labeled data moves clusters with closer pair-wise similarities (likely of the same class) together in the latent space. Both SSL methods outperform the supervised baseline for semi-supervised experiments.

In section 4.3, adversarial attacks have shown the susceptibility of SimCLR based models. The underlying hypothesis is that contrastive self-supervised learning (CSL) leads to adversarial susceptibility [16]. As mentioned in the semi-supervised evaluation discussion results, smaller margin between the classes in the latent space leads to higher adversarial susceptibility. In the supervised latent space, all the same class instances are attracted, leading to a larger margin between the classes which induces adversarial robustness.

It is worth noting that the original RotNet paper achieves a higher performance using AlexNet encoder, which has 3x more parameters. Further testing was conducted with a smaller ResNet-18 encoder for both self-supervised learning methods, which revealed that both methods require a relatively complex architecture to learn meaningful representation.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations, June 2020. URL http://arxiv.org/abs/2002.05709. arXiv:2002.05709 [cs, stat].

[2] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised Representation Learning by Predicting Image Rotations, March 2018. URL http://arxiv.org/abs/1803.07728. arXiv:1803.07728 [cs].

[3] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, Avi Schwarzschild, Andrew Gordon Wilson, Jonas Geiping, Quentin Garrido, Pierre Fernandez, Amir Bar, Hamed Pirsiavash, Yann LeCun, and Micah Goldblum. A Cookbook of Self-Supervised Learning, June 2023. URL http://arxiv.org/abs/2304.12210. arXiv:2304.12210 [cs].

[4] S. Chopra, R. Hadsell, and Y. 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, pages 539–546, San Diego, CA, USA, 2005. IEEE. ISBN 978-0-7695-2372-9. doi: 10.1109/CVPR.2005.202. URL `http://ieeexplore.ieee.org/document/1467314/`.

[5] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful Image Colorization, October 2016. URL `http://arxiv.org/abs/1603.08511`. arXiv:1603.08511 [cs].

[6] O. Chapelle, B. Scholkopf, and A. Zien, Eds. Semi-Supervised Learning (Chapelle, O. et al., Eds.; 2006) [Book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, March 2009. ISSN 1045-9227. doi: 10.1109/TNN.2009.2015974. URL `http://ieeexplore.ieee.org/document/4787647/`.

[7] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4L: Self-Supervised Semi-Supervised Learning, July 2019. URL `http://arxiv.org/abs/1905.03670`. arXiv:1905.03670 [cs].

[8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples, March 2015. URL `http://arxiv.org/abs/1412.6572`. arXiv:1412.6572 [cs, stat].

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks, September 2019. URL `http://arxiv.org/abs/1706.06083`. arXiv:1706.06083 [cs, stat].

[10] Houda Bichri, Adil Chergui, and Mustapha Hain. Image Classification with Transfer Learning Using a Custom Dataset: Comparative Study. *Procedia Computer Science*, 220:48–54, January 2023. ISSN 1877-0509. doi: 10.1016/j.procs.2023.03.009. URL `https://www.sciencedirect.com/science/article/pii/S1877050923005446`.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, December 2015. URL `http://arxiv.org/abs/1512.03385`. arXiv:1512.03385 [cs].

[12] Min Lin, Qiang Chen, and Shuicheng Yan. Network In Network, March 2014. URL `http://arxiv.org/abs/1312.4400`. arXiv:1312.4400 [cs].

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL `https://papers.nips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html`.

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

[15] Adam Coates, Andrew Ng, and Honglak Lee. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, June 2011. URL `http://cs.stanford.edu/~acoates/stl10`. ISSN: 1938-7228.

[16] Rohit Gupta, Naveed Akhtar, Ajmal Mian, and Mubarak Shah. Contrastive Self-Supervised Learning Leads to Higher Adversarial Susceptibility, November 2022. URL `http://arxiv.org/abs/2207.10862`. arXiv:2207.10862 [cs].