solo-learn: A Library of Self-supervised Methods for Visual Representation Learning

Victor G. Turrisi da Costa∗
University of Trento - Trento, Italy

Enrico Fini∗
University of Trento - Trento, Italy

Moin Nabi
SAP AI Research - Berlin, Germany

Nicu Sebe
University of Trento - Trento, Italy

Elisa Ricci
University of Trento and Fondazione Bruno Kessler - Trento, Italy

Editor: Alexandre Gramfort

Abstract

This paper presents solo-learn, a library of self-supervised methods for visual representation learning. Implemented in Python, using Pytorch and Pytorch lightning, the library fits both research and industry needs by featuring distributed training pipelines with mixed-precision, faster data loading via Nvidia DALI, online linear evaluation for better prototyping, and many additional training tricks. Our goal is to provide an easy-to-use library comprising a large amount of Self-supervised Learning (SSL) methods, that can be easily extended and fine-tuned by the community. solo-learn opens up avenues for exploiting large-budget SSL solutions on inexpensive smaller infrastructures and seeks to democratize SSL by making it accessible to all. The source code is available at https://github.com/vturrisi/solo-learn

Keywords: Self-supervised methods, contrastive learning

1. Introduction

Deep networks trained with large annotated datasets have shown stunning capabilities in the context of computer vision. However, the need for human supervision is a strong limiting factor. Unsupervised learning aims to mitigate this issue by training models from unlabeled datasets. The most prominent paradigm for unsupervised visual representation learning is Self-supervised Learning (SSL), where the intrinsic structure of the data provides supervision for the model. Recently, the scientific community devised increasingly effective SSL methods that match or surpass the performance of supervised methods. Nonetheless, implementing and reproducing such works turns out to be complicated. Official repositories of state-of-the-art SSL methods have very heterogeneous implementations or no implementation at all. Although a few SSL libraries (Goyal et al., 2021; Susmelj et al., 2020) are available, they assume that larger-scale infrastructures are available or they lack some recent methods. When approaching SSL, it is hard to find a platform for experiments that

∗. Victor G. Turrisi da Costa and Enrico Fini contributed equally.

©2022 Victor G. Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/ Attribution requirements are provided at http://jmlr.org/papers/v23/21-1155.html
allows running all current state of the art methods with low engineering effort and at the same time is effective and straightforward to train. This is especially problematic because, while the SSL methods seem simple on paper, replication of published results can involve a huge time and effort from researchers. Sometimes official implementations of SSL methods are available, however, releasing standalone packages (often incompatible with each other) is not sufficient for the fast-paced progress in research and emerging real-world applications. There is no toolbox offering a genuine off-the-shelf catalog of state-of-the-art SSL techniques that is computationally efficient, which is essential for in-the-wild experimentation.

To address these problems, we present solo-learn, an open-source framework that provides standardized implementations for a large number of state-of-the-art SSL methods. We believe solo-learn will enable a trustworthy and reproducible comparison between the state of the art methods. The code that powers the library is written in Python, using Pytorch (Paszke et al., 2019) and Pytorch Lightning (PL) (Team, 2019) as back-ends and Nvidia DALI1 for fast data loading, and supports more modern methods than related libraries. The library is highly modular and can be used as a complete pipeline, from training to evaluation, or as standalone modules.

2. The solo-learn Library: An Overview

Currently, we are witnessing an explosion of works on SSL methods for computer vision. Their underlying idea is to unsupervisedly learn feature representations by enforcing similar feature representations across multiple views from the same image while enforcing diverse representations for other images. To help researchers have a common testbed for reproducing different results, we present solo-learn, which is a library of self-supervised methods for visual representation learning. The library is implemented in Pytorch, providing state-of-the-art self-supervised methods, distributed training pipelines with mixed-precision, faster data loading, online linear evaluation for better prototyping, and many other training strategies and tricks presented in recent papers. We also provide an easy way to use the pre-trained models for object detection, via DetectronV2 (Wu et al., 2019). Our goal is to provide an easy-to-use library that can be easily extended by the community, while also including additional features that make it easier for researchers and practitioners to train on smaller infrastructures.

2.1 Self-supervised Learning Methods

We implemented 13 state-of-the-art methods, namely, Barlow Twins (Zbontar et al., 2021), BYOL (Grill et al., 2020), DeepCluster V2 (Caron et al., 2020), DINO (Caron et al., 2021), MoCo V2+ (Chen et al., 2020b), NNCLR (Dwibedi et al., 2021), ReSSL (Zheng et al., 2021), SimCLR (Chen et al., 2020), Supervised Contrastive Learning (Khosla et al., 2020), SimSiam (Chen and He, 2021), SwAV (Caron et al., 2020), VICReg (Bardes et al., 2021) and W-MSE (Ermolov et al., 2021).

2.2 Architecture

In Figure 1, we present an overview of how a training pipeline with solo-learn is carried out. In the bottom, we show the packages and external data at each step, while at the top, we show all the defined variables on the left and an example of the newest defined

1. https://github.com/NVIDIA/DALI
2.3 Comparison to Related Libraries

The most related libraries to ours are VISSL (Goyal et al., 2021) and Lightly (Susmelj et al., 2020), which lack some of our key features. First, we support more modern SSL methods, such as BYOL, NNCLR, SimSiam, VICReg, W-MSE and others. Second, we target researchers with fewer resources, namely from 1 to 8 GPUs, allowing much faster data loading via DALI. Lastly, we provide additional utilities, such as automatic linear evaluation, support to custom datasets and automatically generating UMAP (McInnes et al., 2020) visualizations of the features during training.

3. Experiments

**Benchmarks.** We benchmarked the available SSL methods on CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100 (Krizhevsky et al., 2009) and ImageNet-100 (Deng et al., 2009) and made public the pretrained checkpoints. For Barlow Twins, BYOL, MoCo V2+, NNCLR, SimCLR and VICReg, hyperparameters were heavily tuned, reaching higher performance than reported on original papers or third-party results. Tab. 1 presents the top-1 and top-5 accuracy values for the online linear evaluation. For ImageNet-100, traditional offline linear evaluation is also reported. We also compare with the results reported by Lightly in Tab. 3.
Nvidia DALI vs traditional data loading. We compared the training speeds and memory usage of using traditional data loading via Pytorch Vision\(^2\) against data loading with DALI. For consistency, we ran three different methods (Barlow Twins, BYOL and NNCLR) for 20 epochs on ImageNet-100. Tab. 2 presents these results.

Table 1: Online linear evaluation accuracy on CIFAR-10, CIFAR-100 and ImageNet-100. In brackets, offline linear evaluation accuracy is also reported for ImageNet-100.

| Method        | CIFAR-10 | CIFAR-100 | ImageNet-100 |
|---------------|----------|-----------|--------------|
|               | Acc@1    | Acc@5     | Acc@1        | Acc@5        | Acc@1        | Acc@5        |
| Barlow Twins  | 92.10    | 99.73     | 70.90        | 91.91        | 80.38 (80.16) | 95.28 (95.14) |
| BYOL          | 92.58    | 99.79     | 70.46        | 91.96        | 80.16 (80.32) | 94.80 (94.94) |
| DeepCluster V2| 88.85    | 99.58     | 63.61        | 88.09        | 75.36 (75.40) | 93.22 (93.10) |
| DINO          | 89.52    | 99.71     | 66.76        | 90.34        | 74.84 (74.92) | 92.92 (92.78) |
| MoCo V2+      | 92.94    | 99.79     | 69.89        | 91.65        | 78.20 (79.28) | 95.50 (95.18) |
| NNCLR         | 91.88    | 99.78     | 69.62        | 91.52        | 79.80 (80.16) | 95.28 (95.28) |
| ReSSSL        | 90.63    | 99.62     | 65.92        | 89.73        | 76.92 (78.48) | 94.20 (94.24) |
| SimCLR        | 90.74    | 99.75     | 65.78        | 89.04        | 77.04 (77.48) | 94.02 (93.42) |
| SimSiam       | 90.51    | 99.72     | 66.04        | 89.62        | 74.54 (78.72) | 93.16 (94.78) |
| SwAV          | 89.17    | 99.68     | 64.88        | 88.78        | 74.04 (74.28) | 92.70 (92.84) |
| ViCReg        | 92.07    | 99.74     | 68.54        | 90.83        | 79.22 (79.40) | 95.06 (95.02) |
| W-MSE         | 88.67    | 99.68     | 61.33        | 87.26        | 67.60 (69.06) | 90.94 (91.22) |

Table 2: Speed and memory comparison with and without DALI on ImageNet-100.

| Method     | DALI | 20 epochs | 1 epoch | Speedup | Memory |
|------------|------|-----------|---------|---------|--------|
| Barlow Twins | ✔   | 1h 38m 27s | 4m 55s | -       | 5097 MB |
| BYOL       | ✔   | 1h 38m 46s | 4m 56s | -       | 5409 MB |
| NNCLR      | ✔   | 1h 38m 30s | 4m 55s | -       | 5060 MB |

Table 3: Comparison with Lightly on CIFAR10.

| Method     | Ours | Lightly |
|------------|------|---------|
| SimCLR     | 90.74 | 89.0    |
| MoCoV2+    | 92.94 | 90.0    |
| SimSiam    | 90.51 | 91.0    |

4. Conclusion

Here, we presented solo-learn, a library of self-supervised methods for visual representation learning, providing state-of-the-art self-supervised methods in Pytorch. The library supports distributed training, fast data loading and provides many utilities for the end-user, such as online linear evaluation for better prototyping and faster development, many training tricks, and visualization techniques. We are continuously adding new SSL methods, improving usability, documents, and tutorials. Finally, we welcome contributors to help us at https://github.com/vturrisi/solo-learn

Acknowledgments. This work was supported by a joint project under Grant No. JQ18012, by the EU H2020 AI4Media No. 951911 project and the European Institute of Innovation & Technology (EIT).

\(^2\) https://github.com/pytorch/vision
References
Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. arXiv:2105.04906, 2021.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In NeurIPS, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. arXiv:2104.14294, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In ICML, 2020a.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In CVPR, 2021.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. ICLR, 2021.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. arXiv:2104.14548, 2021.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In ICML, 2021.

Priya Goyal, Quentin Duval, Jeremy Reizenstein, Matthew Leavitt, Min Xu, Benjamin Lefaudeux, Mannat Singh, Vinicius Reis, Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Ishan Misra. Vissl. GitHub. Note: https://github.com/facebookresearch/vissl, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In NeurIPS, 2020.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. NeurIPS, 2020.
Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *International Conference on Computer Vision (ICCV)*, 2021.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv:1802.03426*, 2020.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Igor Susmelj, Matthias Heller, Philipp Wirth, Jeremy Prescott, and Malte Ebner et al. Lightly. *GitHub*. Note: https://github.com/lightly-ai/lightly, 2020.

Pytorch Lightning Development Team. Pytorch lightning. *GitHub*. Note: https://github.com/PyTorchLightning/pytorch-lightning, 3, 2019.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

Jure Zbontar, Li Jing, Ishan Misra, Yann Lecun, and Stephane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

Mingkai Zheng, Shan You, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Ressl: Relational self-supervised learning with weak augmentation. *arXiv:2107.09282*, 2021.