Self-Supervised Relational Reasoning for Representation Learning
(2020)

Massimiliano Patacchiola, Amos Storkey

Summary

Contributions

In order to learn useful representation from unlabeled data, self-supervised learning consists of defining some pretext tasks generated in an unsupervised manner, and then used to guide the model during training and build useful representations that can be used in downstream task. In this paper, the authors propose a new self-supervised pre-text task based on the relational learning paradigm. Relational reasoning is based on a simple design principle: the use of a relation network as a learnable function to quantify the relationships between a set of objects. Based on this, we can use relational reasoning as a pretext task by training the relation head on unlabeled data focusing on relations between the views of the same object generated using standard augmentation and relation between different objects in different scenes.

Method

In the standard self-supervised pipeline, given a model consisting of a back-bone and a head, we first train them on a pre-text task and then head is discarded and the backbone is transfered to reused for downstream tasks (eg image classification or segmentation). In order to adapt the relational reasoning paradigm for self-supervised leanring, the first step is to define the pre-text task. Let a set of objects $\mathcal{O} = \{o_1, \ldots, o_N\}$ of the same scene, our objective is to train a learner to be able to differentiate between objects from every scene possible. To this end the training objective is to discriminate between two objects, and the learner needs to predict if two object are similar, ie, $\{o_i, o_j\} \rightarrow$ same belong to the same category, or are different, ie, $\{o_i, o_j\} \rightarrow$ different. The second step is to define data augmentations to (1) construct the positives (two views of the same object) and (2) making between-scenes reasoning more complicated and force the model to learn useful representations. So for a given object o_i, two create two views of it where the model needs to predict the same category, we apply two random augmentation $\mathcal{A}$ of the same object $\{\mathcal{A}(o_i), \mathcal{A}(o_i)\} \rightarrow$ same. As for the negative case, we samply apply two augmentation two to different objects sampled randomly from the dataset $\left\{\mathcal{A}(o_i), \mathcal{A}\left(o_{\setminus i}\right)\right\} \rightarrow$ different.

Loss Function. After introducing our pre-text task and the process of generating the positive and negative pairs, we now need to formulate the task into a loss function in order to train the model. Let the unlabeled dataset be $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}$ and our back-bone be $f_\theta(\cdot)$ that generate representation $f_\theta(\mathbf{x}_n) = \mathbf{z}_n$ that are collected into a set $\mathcal{Z} = \{\mathbf{z}_n\}_{n=1}^{N}$. And let $r_\phi(\cdot)$ be a relation module, taking as input an aggregate $g(.,.)$ (such as sum, max or concatenation) of a pair of representation and return a score y and the binary loss $\mathcal{L}(y, t)$ as the loss between the score y and the target t. The complete training objective can be specified as

$$\underset{\boldsymbol{\theta},\boldsymbol{\phi}}{\operatorname{argmin}} \sum_{n=1}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K} \underbrace{\mathcal{L}\left(r_\phi\left(a\left(\mathbf{z}_n^{(i)}, \mathbf{z}_n^{(j)}\right)\right), t=1\right)}_{\text{intra-reasoning}} + \underbrace{\mathcal{L}\left(r_\phi\left(a\left(\mathbf{z}_n^{(i)}, \mathbf{z}_{\setminus n}^{(j)}\right)\right), t=0\right)}_{\text{inter-reasoning}}, \text{ with } \mathbf{z}_n = f_\theta(\mathbf{x}_n),$$

where we simply task the learner with predicting 1 over all pairs of augmentations of the same inputs, and 0 for the rest of the pairs. Specifically, the learning objective is a binary classification over P representation pairs, where the score y represents the probability of the input pairs of being a depiction of the different objects

$$\mathcal{L}(\mathbf{y}, \mathbf{t}, \gamma) = \frac{1}{P} \sum_{i=1}^{P} -w_i \left[t_i \cdot \log y_i + (1 - t_i) \cdot \log (1 - y_i) \right]$$

with an optional weight scaling factor w_i that gives more importances to uncertain estimations. To summarize the process is depicted bellow.

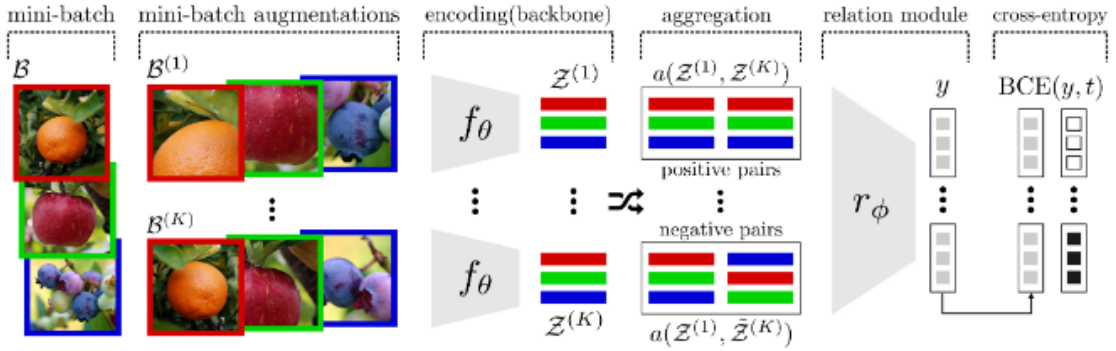
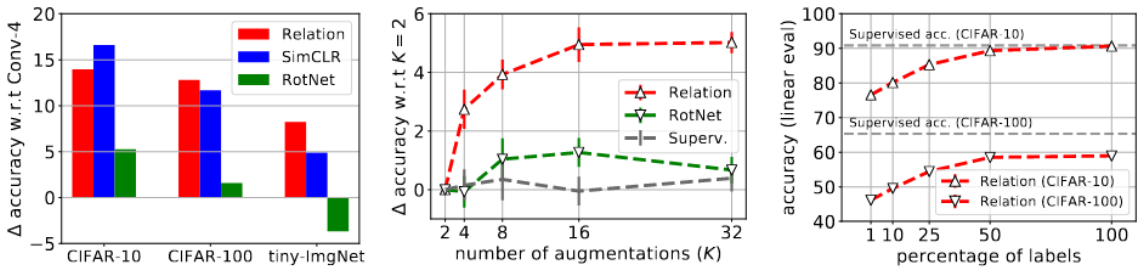


Figure 1: Overview of the proposed method. The mini-batch $\mathcal{B}$ is augmented K times (e.g. via random flip and crop-resize) and passed through a neural network backbone f_θ to produce the representations $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(K)}$. An aggregation function a joins positives (representations of the same images) and negatives (randomly paired representations) through a commutative operator. The relation module r_ϕ estimates the relational score y, which must be 1 for positives and 0 for negatives. The model is optimized minimizing the binary cross-entropy (BCE) between prediction and target t.

Results

Table 1: Comparison on various benchmarks. Mean accuracy (percentage) and standard deviation over three runs (ResNet-32). Best results in bold. **Linear Evaluation:** training on unlabeled data and linear evaluation on labeled data. **Domain Transfer:** training on unlabeled CIFAR-10 and linear evaluation on labeled CIFAR-100 (10→100), and viceversa (100→10). **Grain:** training on unlabeled CIFAR-100, linear evaluation on coarse-grained CIFAR-100-20 (20 super-classes). **Finetune:** training on the unlabeled set of STL-10, finetuning on the labeled set (ResNet-34).

	Linear Evaluation		**Domain Transfer**		**Grain**	**Finetune**
Method	**CIFAR-100**	**tiny-ImgNet**	**10→100**	**100→10**	**CIFAR-100-20**	**STL-10**
Supervised (upper bound)	65.32±0.22	50.09±0.32	33.98±0.71	71.01±0.44	76.35±0.57	69.82±3.36
Random Weights (lower bound)	7.65±0.44	3.24±0.43	7.65±0.44	27.47±0.83	16.56±0.48	n/a
DeepCluster (Caron et al., 2018)	20.44±0.80	11.64±0.21	18.37±0.41	43.39±1.84	29.49±1.36	73.37±0.55
RotationNet (Gidaris et al., 2018)	29.02±0.18	14.73±0.48	27.02±0.20	52.22±0.70	40.45±0.39	83.29±0.44
Deep InfoMax (Hjelm et al., 2019)	24.07±0.05	17.51±0.15	23.73±0.04	45.05±0.24	33.92±0.34	76.03±0.37
SimCLR (Chen et al., 2020)	42.13±0.35	25.79±0.35	36.20±0.16	65.59±0.76	51.88±0.48	89.31±0.14
Relational Reasoning (ours)	**46.17±0.17**	**30.54±0.42**	**41.50±0.35**	**67.81±0.42**	**52.44±0.47**	**89.67±0.33**



(a) (b) (c)

Figure 2: (a) Difference in accuracy using the deeper backbone (Conv4→ResNet-32, linear evaluation). As the complexity of the dataset raises our method performs increasingly better than the others. (b) Correlation between validation accuracy (3 seeds, Conv-4, CIFAR-10) and number of mini-batch augmentations. Only in our method the accuracy is positively correlated with the number of augmentations. (c) Semi-supervised accuracy with an increasing percentage of labels (ResNet-32).

Table 2: Linear evaluation. Self-supervised training on unlabeled data and linear evaluation on labeled data. Comparison between three datasets (CIFAR-10, CIFAR-100, tiny-ImageNet) for a shallow (Conv-4) and a deep (ResNet-32) backbone. Mean accuracy (percentage) and standard deviation over three runs. Best results highlighted in bold.

	Conv-4			**ResNet-32**		
Method	**CIFAR-10**	**CIFAR-100**	**tiny-ImageNet**	**CIFAR-10**	**CIFAR-100**	**tiny-ImageNet**
Supervised (upper bound)	80.46±0.39	49.29±0.85	36.47±0.36	90.87±0.41	65.32±0.22	50.09±0.32
Random Weights (lower bound)	32.92±1.88	10.79±0.59	6.19±0.13	27.47±0.83	7.65±0.44	3.24±0.43
DeepCluster (Caron et al., 2018)	42.88±0.21	21.03±1.56	12.60±1.23	43.31±0.62	20.44±0.80	11.64±0.21
RotationNet (Gidaris et al., 2018)	56.73±1.71	27.45±0.80	18.40±0.95	62.00±0.79	29.02±0.18	14.73±0.48
Deep InfoMax (Hjelm et al., 2019)	44.60±0.27	22.74±0.21	14.19±0.13	47.13±0.45	24.07±0.05	17.51±0.15
SimCLR (Chen et al., 2020)	60.43±0.26	30.45±0.41	20.90±0.15	**77.02±0.64**	42.13±0.35	25.79±0.40
Relational Reasoning (ours)	**61.03±0.23**	**33.38±1.02**	**22.31±0.19**	74.99±0.07	**46.17±0.16**	**30.54±0.42**

Table 3: Linear evaluation on SlimageNet (Antoniou et al., 2020). This dataset is more challenging than ImageNet, since it only has 160 low-resolution (64×64) color images for each one of the 1000 classes of ImageNet. Below is reported the linear evaluation accuracy on labeled data with a ResNet-32 backbone, after training on unlabeled data. Mean accuracy (percentage) and standard deviation over three runs. Best result highlighted in bold.

Method	**SlimageNet**
Supervised (upper bound)	33.94±0.21
Random Weights (lower bound)	0.79±0.09
RotationNet (Gidaris et al., 2018)	7.25±0.28
SimCLR (Chen et al., 2020)	14.32±0.24
Relational Reasoning (ours)	**15.81±0.72**