Saliency Can Be All You Need in Contrastive Self-supervised Learning

Veysel Kocaman[1(✉)], Ofer M. Shir[2], Thomas Bäck[1], and Ahmed Nabil Belbachir[3]

[1] LIACS, Leiden University, Leiden, The Netherlands
v.kocaman@liacs.leidenuniv.nl
[2] Computer Science Department, Tel-Hai College and Migal Institute, Upper Galilee, Israel
[3] Technology, NORCE Norwegian Research Centre, Grimstad, Norway

Abstract. We propose an augmentation policy for Contrastive Self-Supervised Learning (SSL) in the form of an already established Salient Image Segmentation technique entitled Global Contrast based Salient Region Detection. This detection technique, which had been devised for unrelated Computer Vision tasks, was empirically observed to play the role of an augmentation facilitator within the SSL protocol. This observation is rooted in our practical attempts to learn, by SSL-fashion, aerial imagery of solar panels, which exhibit challenging boundary patterns. Upon the successful integration of this technique on our problem domain, we formulated a generalized procedure and conducted a comprehensive, systematic performance assessment with various Contrastive SSL algorithms subject to standard augmentation techniques. This evaluation, which was conducted across multiple datasets, indicated that the proposed technique indeed contributes to SSL. We hypothesize whether salient image segmentation may suffice as the only augmentation policy in Contrastive SSL when treating downstream segmentation tasks.

1 Introduction

Despite recent advances in Computer Vision (CV), the effectiveness of visual recognition and learning still very much depends on manual annotations and on how well they represent the images at hand. Given the substantial amount of available unlabeled data and the costly manual annotation, new methods for online and unsupervised learning are crucial for achieving robust and efficient visual learning. Current unsupervised learning methods do not perform learning under a genuine unsupervised state – they rather focus on transfer learning, or unsupervised tasks subject to strong features that were pre-trained in a supervised way. Therefore, there is a need to investigate algorithms for unsupervised learning of completely new categories, such as self-supervised learning (SSL) to learn in a continuous, long-term fashion. Indeed, SSL needs also to generalize the classical unsupervised learning scenario of physical objects to scenarios of

G. Bebis et al. (Eds.): ISVC 2022, LNCS 13599, pp. 119–140, 2022.
https://doi.org/10.1007/978-3-031-20716-7_10

higher-level actions and more complex activities, and at the same time develop capabilities to detect abnormalities in visual data [1,2].

The requirement for very large datasets of manually labeled instances may seem counter-intuitive since this is not how humans learn to recognize new objects. Humans are constantly fed with images through their eyes, and are able to learn an object's appearance and to distinguish it from other objects without knowing what the object exactly is. Moreover, collecting large-scale datasets is time-consuming and expensive, while the supervised approach to learn features from labeled data has almost reached its saturation due to the intense labor required in manually annotating millions of data instances. This is because most of the modern CV systems (that are supervised) try to learn some form of image representations by finding a pattern that links the data points to their respective annotations in large datasets [3]. Moreover, the data annotation efforts vary from task to task, and it is estimated that the time spent on image segmentation and object detection (i.e., carefully drawing boundaries) is four times longer than the image classification itself [4]. The annotation efforts become significantly higher when it comes to highly regulated and specialized domains like medicine and finance in which the expertise level of the human annotator matters more than in any other domain. Moreover, supervised learning not only depends on expensive annotations but also suffers from other drawbacks such as generalization errors, spurious correlations, and being prone to adversarial attacks [5].

In this study, we explore the viability of using an unsupervised image segmentation technique called 'Global Contrast based Salient Region Detection (SGD)' [6] as an augmentation policy (rather than a proxy task) in contrastive SSL methods and study its impact on downstream supervised image segmentation tasks in low data regimes. To the best of our knowledge, this is the first study that explores salient object segmentation techniques as an augmentation policy in contrastive SSL.

We will show that using this SGD algorithm as an image augmentation policy in SSL produces better representations for downstream image segmentation tasks when compared to default augmentation policies commonly utilized in SSL methods. In order to make the integration of SGD into SSL pretraining routines feasible, we also devise a simple manipulation called offline augmentation with hashing that enables running comprehensive experiments with various parameters and configurations. We will also provide evidence that SGD-based augmentation policy in SSL performs better with low resolution images.

The current study targets the following *research questions*:

> Would SGD produce better representation when used as an augmentation policy in Contrastive SSL? Moreover, would salient image segmentation suffice as the only augmentation policy in Contrastive SSL when treating downstream segmentation tasks?

The concrete contributions of this paper are the following:

- Comparing a relatively old unsupervised image segmentation technique, Global Contrast based Salient Region Detection (SGD), with recent deep learning (DL)-based image segmentation algorithms.

- Evaluating the viability of a generative model (Pix2Pix) as a proxy for computationally expensive image augmentation methods.
- Devising an SGD-based efficient offline augmentation technique to incorporate any expensive augmentation policy in DL training routines.
- Employing SGD as an offline augmentation policy (rather than a proxy task) in contrastive SSL methods and study its impact on downstream supervised image segmentation and object detection tasks.
- Illustrating that fine-grained details in high resolution images would negatively impact the performance of SSL when compared to the coarse-grained details in low resolution images.
- Formulating a recommendation for choosing the most appropriate SSL method (accounting for the augmentation technique, downstream task and even the imagery resolution).

The remainder of the paper is organized as follows: Sect. 2 describes the concrete motivation that ignited this research, and then summarizes related work as well as various SSL approaches, including the role of data augmentation therein. It concludes with presenting the SGD method in detail. Section 3 outlines the experimental setup regarding SGD, including the datasets the various preliminary efforts to make SGD-based augmentation policies viable in DL training routines. It then presents the attained results. Section 4 discusses the findings and proposes possible *mechanistic* explanations, and finally Sect. 5 concludes by pointing out the key points and future directions.

2 Motivation and Background

2.1 Related Work

Semantic segmentation is the task of assigning each pixel to a specific class label. The class labels can be the same as for object detection, but unlike the object detection task, which labels each instance of an object as separate objects, semantic segmentation only assigns a pixel a specific class label and does not differentiate instances of objects.

As the augmentations in any SSL method should be tailored *a priori* and fit in terms of the downstream tasks, devising an augmentation policy for downstream segmentation tasks depends on harnessing the intrinsic features that help model learn how to segment the objects in an image. This is usually achieved through auxilary tasks such as rotating, cropping, colorization etc. One of the most useful auxilary tasks can be regarded as image colorization that is introduced in [7] as a process of estimating RGB colors for grayscale images. The backbone network within the pretrained model performed well for downstream tasks like object classification, detection, and segmentation compared to other methods. The study in [8] also suggested a similar technique to automatically colorize grayscale images by merging local information dependent on small image patches with global priors computed using the entire image. Since these two techniques employ a strategy of finding suitable reference images and transferring

their color onto a target grayscale image, the semantic information plays a little role and has the potential to actually hamper the SSL. Probably one of the most useful colorization tasks is suggested by [9], in which a system must interpret the semantic composition of the scene (what is in the image) as well as to localize objects (where things are) to incorporate semantic parsing and localization into a colorization system. In a generative manner, [10] proposed image inpainting, a context-based pixel prediction where the network understands the context of the entire image as well as the hypothesis for missing parts, hence assisting in extracting the semantic information.

There are many other auxiliary tasks proposed in SSL but most of these methods focus on basic inherent visual features, like image patches and rotation, which are very simple tasks that are not likely to completely learn the semantics and the spatial features of the image. Actually, the default augmentation policies in contrastive SSL methods are the derivation of these auxiliary tasks. To the best of our knowledge, salient image segmentation has not been used as an auxiliary task in any SSL method before, let alone contrastive SSL.

2.2 Concrete Background

To avoid time-consuming and expensive data annotations, as an alternative to data-hungry supervised learning methods, many SSL methods were proposed to learn visual features from large-scale unlabeled images or videos without using any human annotations. Formally, SSL is a subset of unsupervised learning methods, referring to learning methods in which ConvNets are explicitly trained with automatically generated labels; so the supervision comes from structure of the data itself. Since no human annotations are needed to generate pseudo labels during self-supervised training, very large-scale datasets can be used for SSL training. Trained with these pseudo labels, self-supervised methods typically achieve promising results, while gradually closing the performance gap with respect to supervised methods [3].

On the other hand, in order to address the lack of annotated datasets or dataset shifts, there are various efforts towards developing zero-shot or few-shot learners. Zero-shot learners (ZSL) aim to predict the correct class without being exposed to any instances belonging to that class in the training dataset, while few-shot learners (FSL) attempt to accomplish the same when a small number of examples are available in the training dataset. In short, ZSL/FSL and SSL have important commonalities in that they can be applied in situations where annotated data is scarce. This close relationship allows some of the SSL methods to gain semantic scene understanding through a pretraining process. For example, certain attention heads in DINO [11] are found to be discovering and segmenting objects in an image or a video with no supervision and without being given a segmentation-targeted objective. However, it is a computationally expensive process to segment an image using transformer based architectures and usually this does not work well.

Another related concept, entitled semi-supervised learning, refers to a learning problem involving a small portion of labeled examples and a large number

of unlabeled examples from which a model must learn and make predictions on new examples. It is halfway between supervised and unsupervised learning, and directly relevant to a multitude of practical problems where it is relatively expensive to produce labeled data. Some of the most popular semi-supervised learning methods include pseudo learning [12] (a model is trained on a labeled dataset and used to predict pseudo-labels for the unlabeled data) and a noisy student [13] (training two separate models called "Teacher" and "Student"). Being central to our experiments, we will focus more on SSL and SGD in this section.

2.2.1 Self-Supervised Learning (SSL)

Explicitly, SSL is a machine learning process where the model trains itself to learn one part of the input from another part of the input. In other words, the model learns from labels that are presumably already intrinsic in the data itself. This process is also known as predictive or pretext learning and the unsupervised problem is transformed into a supervised problem by auto-generating the labels. To effectively exploit the huge quantity of unlabeled data, it is crucial to set the right learning objectives, in order to obtain the appropriate supervision from the data itself.

Within the context of utilizing unlabeled data to learn the underlying representations, SSL can be organized as (i) handcrafted pretext tasks-based, (ii) contrastive learning-based and (iii) clustering learning-based approaches. In handcrafted pretext tasks-based method, a popular approach has been to propose various pretext tasks that help in learning features using pseudo-labels while the networks can be trained by learning objective functions of the pretext tasks and the features are learned through this process [3]. Tasks such as image-inpainting [10], colorizing gray-scale images [7], solving jigsaw puzzles [14], image super-resolution [15], video frame prediction [16], audio-visual correspondence [17], to mention the most prominent, have proven to be effective for learning good representations. In doing so, the model learns quality representations of the samples and is used later for transferring knowledge to downstream tasks. The selection of an appropriate proxy task (pretext) is critical to the effectiveness of self-supervised learning. It requires careful design, and indeed, numerous researchers investigated various approaches for given downstream tasks. For example, [18] proposed a proxy task to classify whether a given pair of kidneys belong to the same side; with the assumption that the network needs to develop an understanding of the structure and sizes of the kidneys.

On the other hand, contrastive learning (CL) is a training method wherein a classifier distinguishes between "similar" (positive) and "dissimilar" (negative) input pairs. It is essentially the task of grouping similar samples closer to each other, unlike setting diverse samples far from each other. During training, the augmented version of the original sample is considered as a positive sample, and the rest of the samples in the batch/dataset (depends on the method being used) are considered negative samples. Next, the model is trained in a way that it learns to differentiate positive samples from the negative ones. Constructing

positive and negative pairs via data augmentation allows the model to learn from inter-class variance (uniformity) by pushing negative pairs far away, and from intra-class similarity (alignment) by pulling positive pairs together. The core concept is to maximize the dot product between the feature vectors which are similar and minimize the dot product between those of which are not similar. CL methods use contrastive loss that is evaluated based on the feature representations of the images extracted from an encoder network. The popular methods that recently started to produce results comparable to the state-of-the-art supervised learning methods, even with less labelled images, can be regarded as DINO [11], SwAV [19], MoCo [20,21], BYOL [22], SimSiam [23] and SimCLR, [24,25]. The representation extraction strategies differ from one method to another (e.g. BYOL does not even need negative pairs) but the changes are very subtle and without rigorous ablations, it is hard to tell which one works better on a case at hand. The widely used approach to evaluate the learned representations through the SSL's pre-training process is the linear evaluation protocol [26], where a linear classifier (e.g., SVM, Logistic Regression, etc.) is trained on top of the frozen backbone network (image representations are derived from the final or penultimate layers of the backbone network as a feature vector). Finally, the test accuracy is used as a proxy for representation quality (Fig. 1).

Data augmentation plays an important role in the learning process of contrastive SSL methods, while the composition of multiple data augmentation operations is crucial in defining the contrastive prediction tasks that accomplish effective representations. In addition, contrastive SSL benefits from stronger data augmentation, when compared to supervised learning. Augmentations can be regarded as an indirect way to pass human prior knowledge into the model, and an effective augmentation should discard the unimportant features for the downstream task (i.e., removing the "noise" to classify an image). The nature of the selected augmentations should fit the downstream task, maintain the image semantics (meaning), introduce the model with challenging assignments/tests. Their selection depends upon the dataset's underlying distribution and cardinality, whereas a recent study [27] further claims that augmentations are problem-class-dependent.

The most popular data augmentation techniques in contrastive SSL include rotation, crop, cut, flip, color jitter, blur, Gaussian noise and gray scale. Almost all of the contrastive SSL methods use these (or similar) augmentation techniques at certain degrees no matter what the downstream task is. Selection of the most suitable augmentation policy is a crucial step in contrastive SSL. [25] systematically studied the impact of data augmentation and observed that no single transformation suffices to learn good representations, even though the model can almost perfectly identify the positive pairs in the contrastive task. When composing augmentations, the contrastive prediction task becomes harder, but the quality of representation improves dramatically. Compared to handcrafted pretext tasks-based methods, contrastive SSL methods are easier to implement but the pretraining process is computationally expensive as they require large batches and large amount of unlabelled datasets. Nevertheless, the pretrained

backbone ConvNets through contrastive SSL methods generally produce better latent representations and perform well on downstream tasks.

Importantly, even though classical image segmentation methods can be regarded as one of the pretext tasks in SSL, the downstream segmentation and object detection tasks usually try to detect/segment certain salient objects and areas in an image, rather than entire textures, which unsupervised segmentation algorithms generate. While essentially solving a segmentation problem, salient object detection and segmentation approaches segment only the salient foreground object from the background, rather than partitioning an image into regions of coherent properties as in general segmentation algorithms. That is, using a salient image segmentation in SSL is likely to generate better representations when the downstream task is defined as either image segmentation or object detection.

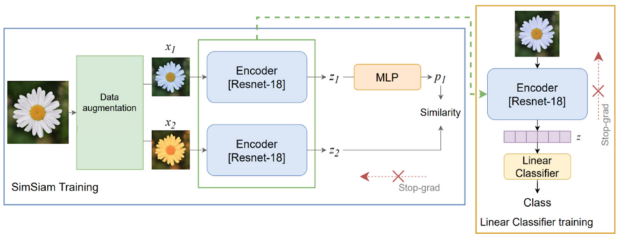

Fig. 1. Linear evaluation protocol to evaluate the learned representations of a pretraining process of SimSiam [23].

2.2.2 Global Contrast Based Salient Region Detection (SGD)

SGD, also known as SaliencyCut [6], is an automated unsupervised salient region extraction method, an improved iterative version of GrabCut [28], which introduced a contrast analysis method to integrate spatial relationships into region-level contrast computation. In GrabCut, the user initially draws a rectangle around the foreground region in an image, and then the algorithm iteratively segments it to get the best result. The executed steps are (i) estimating the color distribution of the foreground and background via a Gaussian Mixture Model (GMM), (ii) constructing a Markov random field over the pixels labels (i.e., foreground vs. background), and (iii) applying a graph cut optimization to arrive at the final segmentation. Instead of manually selecting this rectangular region to initialize the process, SaliencyCut is using histogram-based contrast (HC) and region-based contrast (RC) techniques (that are also suggested in the same paper) to create saliency maps and then binarizes this map using a fixed threshold (e.g., value of 70).

In RC, the input image is first segmented into regions, then the color contrast at the region level is computed, and saliency for each region is finally defined as the weighted sum of the region's contrasts to all other regions in the image.

The weights are set according to the spatial distances with farther regions being assigned smaller weights.

In HC, the number of colors needed to consider is reduced to 1728 by quantizing each color channel to have 12 different values (12 color values per channel in R-G-B). Considering that color in a natural image typically covers only a small portion of the full color space, the number of colors is further reduced by ignoring less frequently occurring colors. By choosing more frequently occurring colors, and by ensuring that they cover the colors used *de facto* by more than 95% of the image pixels, we are typically left with around n = 85 colors. The colors of the remaining pixels, which comprise fewer than 5% of the image pixels, are replaced by the closest colors in the histogram.

Once the saliency map is initialized with RC and HC, GrabCut is iteratively run (i.e., iterative refinements) to improve the SaliencyCut result. After each iteration, dilation and erosion operations are used on the current segmentation result to get a new trimap for the next GrabCut iteration. During this iterative refinement, adaptive fitting is used (regions closer to an initial salient object region are more likely to be part of that salient object than far-away regions). Thus, the new initialization enables GrabCut to include nearby salient regions, and exclude non-salient regions according to color dissimilarity. See [6] for more details regarding SGD. Finally, Fig. 2 presents the outcomes' comparison of SGD-

Fig. 2. Comparing the outcomes of unsupervised versus SGD-based segmentation on PASCAL VOC 2012. Invariant Information Clustering (IIC) [30], Superpixels [31], Continuity Loss [32]. Different segments are shown in different colors (the images in the first 5 rows are taken from [32]). Notably, SGD was developed at least 8 years prior to IIC and Superpixels, and can still perform comparatively better on some cases. (Color figure online)

based segmentation versus latest unsupervised segmentation on PASCAL VOC 2012 [29].

3 Implementation, Setup and Results

Given the aforementioned research question, we derive concrete experimentation tasks:

1. Preliminary: Applying SGD to the PVs' datasets
2. Obtaining a solid and an efficient implementation
3. Testing the primary hypothesis: SGD as an augmentation policy

3.1 Setup and Datasets

We used the following datasets in the current study:

- **Aerial Drone Images for Solar Photovoltaic (PV) Panels Defects (NORCE-PV):** This dataset is provided by NORCE (Norwegian Research Centre) and has 790 high resolution aerial drone images. It is originally gathered for PV panels defect detection classification and annotated by NORCE internally. In this study, we will not be dealing with defect classes but the images themselves to run the experiments. The dataset is not publicly available.
- **Multi-resolution PV Dataset From Satellite and Aerial Imagery (MultiRes-PV):** This dataset includes three groups of PV samples collected at the spatial resolution of 0.8m, 0.3m and 0.1m, namely PV08 from Gaofen-2 and Beijing-2 imagery, PV03 from aerial photography, and PV01 from UAV orthophotos. In this study, we only used PV01 rooftop images (645 in total) that comes with segmentation masks that come within 256×256 size [33].
- **CIFAR10:** The CIFAR-10 dataset consists of 60000 32×32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images [34].
- **STL10:** The STL-10 dataset is a subset of ImageNet, consists of 1300 labelled, 100000 96×96 unlabelled colour images in 10 classes. In particular, each class has fewer labeled training examples than in CIFAR-10, but a very large set of unlabeled examples is provided to learn image models prior to supervised training [35].

3.2 Preliminary: Running SGD on NORCE-PV and MultiRes-PV Datasets

SGD is a computationally expensive process and it produces segmentations on various level of details given the size of an image. For instance, it takes $\sim$2 min to generate a saliency map (segmentation) of an 600×450 RGB image, while it takes around $\sim$ 1 s to accomplish the same process on 60×45 (10% of

the original image). In the cases wherein a highly detailed segmentation map is needed, SGD implementation would be practically impossible to implement on real time. In some other cases wherein a rough segmentation map would suffice, applying SGD on a reduced size of an image would still produce useful segmentation. On the other hand, when we apply SGD on low resolution images like MultiRes-PV images (256×256), we may need to upscale the image to get a better segmentation. The outcome of applying SGD on various sizes of NORCE-PV (high resolution) and MultiRes-PV (low resolution) images is presented in Figs. 4 and 5, respectively.

We also run STEGO [36] and DETIC [37], two recent popular zero-shot DL-based image segmentation algorithms, on our datasets and compare results with SGD. As evident in Fig. 3, DETIC obtains impressive results as it is already capable of detecting solar arrays from natural images (it is trained to detect 20 thousand different objects out of the box). Compared to the heavy STEGO algorithm, developed just a few months ago, SGD performs comparatively better.

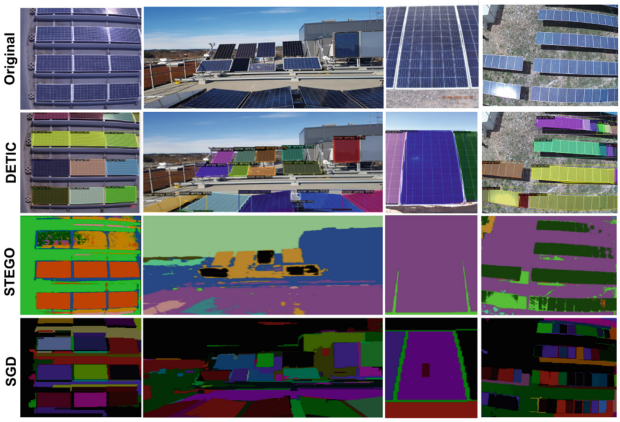

Fig. 3. Comparison of transformer-based zero-shot segmentation methods versus SGD on NORCE PV dataset.

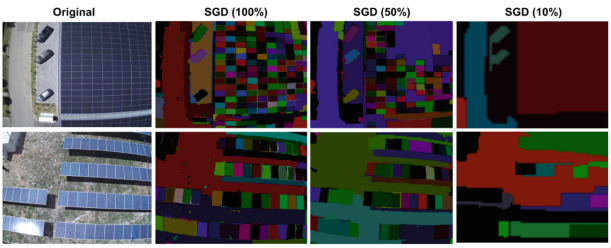

Fig. 4. Segmentation produced by SGD from NORCE-PV images with various sizes.

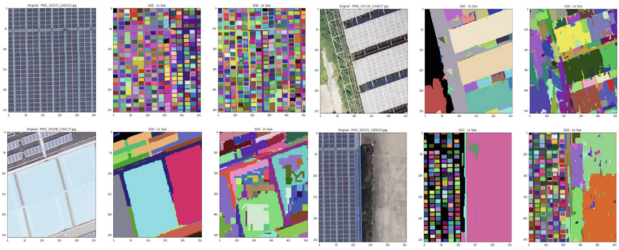

Fig. 5. Segmentation produced by SGD from low resolution MultiRes-PV images with original versus 200% rescaled.

3.3 An Efficient Implementation

Employing SGD on any DL training routine as an augmentation or transformation policy is highly inefficient in terms of computational costs. Notably, using GPU is not an option due to a layered (a mixture of OpenCV and Python functions) image preprocessing techniques along with RC and HC methods. The fact that thousands of images are to pass through this process in each epoch renders the integration of SGD in the DL training process practically infeasible. Our tests indicate that training a Resnet-18 ConvNet with SGD augmentation to classify PV defects using the NORCE-PV dataset (790 images, shape 32×32), for one epoch, takes ∼50 min on Tesla K80 (11.5 GB) GPU.

3.3.1 Image to Segmentation Map Translation via Pix2Pix

To achieve feasiblity, we firstly thought about training a generative DL model to learn SGD segmentations via image translation and then replace the SGD process with this generative model. We chose the Pix2Pix architecture [38] for this purpose and trained several models. We basically fed the original images and SGD-segmented version to the network and trained further to get the model learn translating one image to another. Using this Pix2Pix model within the Resnet-18 training routine to replace SGD reduced the duration from ∼50 min to ∼30 min per epoch. Even though the results look promising (Fig. 6), there are several drawbacks, e.g., information loss during the translation process, being outperformed on image instances that were under-represented, and the limited speed-improvement gains – which altogether render this process unviable in practice.

3.3.2 Offline Augmentation with Hashing

SGD is a computationally expensive method that results in a speed bottleneck when used in DL training routines and it might be one of the reasons that such a strong segmentation method could not establish itself despite all the latest advancements in similar fields. Since SGD is basically a deterministic approach, the segmentation map is always the same except colors; hence the contours and overall shapes are always identical for an image. In other words, even if the

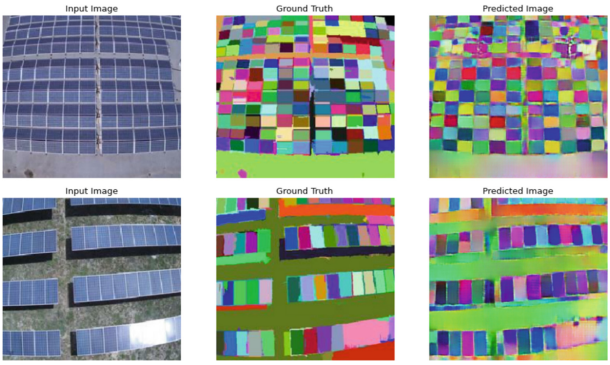

Fig. 6. The second column of this figure shows the SGD segmentation of the original images and the images generated with Pix2Pix image translation model are shown at the third column.

pixel colors change at each iteration of SGD, the overall shapes and lines in an image are always the same, unless resizing is applied. Experimenting around this attribute, we devised an unprecedented solution by running SGD once for all the images in the dataset and creating a segmentation mask for each image and then reusing that every epoch during training without running SGD again and again. In order to simulate the random colors and make the model invariant to colors (so it can focus more on other features, e.g., contrast, shapes, lines etc.), we also applied random color jittering and swapping from the same color palette utilized by SGD. Next, we describe the explicit steps:

1. Read every image in the dataset as a `numpy` array and hash it to store the entire image as a single hash code (string),
2. Run SGD over every image in the dataset and save the segmentation map as an image to the disk,
3. Create a dictionary (key-value pairs) with the hash strings (of Step-1) as keys and the file path of every segmentation map as the associated value (if reading from disk becomes a bottleneck, all of the segmentation maps can be read at once and stored in the memory as an array),
4. In model training with original images, get the hashmap of every image array and find the file path of corresponding segmentation map from the dictionary above at each iteration, and read that image from the disk or from memory if it is stored in memory,
5. Apply color jittering to randomly swap the colors of the augmented image and use that as a proxy augmented image during the rest of the process.

This technique allowed SGD process to run instantly (practically below 1sec) as the segmentation maps are read from disk/memory at real-time at no cost.

3.4 Using SGD as an Augmentation Policy in Contrastive SSL Algorithms

In the Contrastive SSL pre-training routine, the augmented version of the original sample is considered as a positive sample, and the rest of the samples in the batch/dataset (depends on the method being used) are considered negative samples. Then, the model is trained in a way that it learns to differentiate positive samples from the negative ones. In doing so, the model learns quality representations of the samples and is used later for transferring knowledge to downstream tasks.

A predefined CV architecture (e.g., ResNet50, VGG19, EfficientNet etc.) is typically used as a ConvNet backbone, and the weights (model parameters) are updated during this contrastive SSL process. Then, the backbone is saved and used in another architecture as a starting point or just as a facilitator of feature extraction (representations from one of the last fully connected (FC) layers).

In this study, in order to quickly iterate across various heavy augmentation combinations under limited computational resources, we used a lightweight ResNet18 architecture and then pretrained several backbones with the combination of SGD versus standard image augmentation techniques (crop, grayscaling, jittering etc.) using the following SSL algorithms: SimSiam [23], BYOL [22], SimCLR [25], MoCo [20], SwAV [19] and Barlow Twins [39]. Then we tested the effectiveness of these backbones in downstream image clustering, image segmentation, object detection and classification problems.

3.4.1 Image Clustering Using SSL Backbones

Using the 80% of the entire NORCE-PV dataset with no label, we trained SvAW, SimSiam and SimCLR models for 100 epochs with the combinations of default SSL augmentations and SGD (with offline augmentation through hashing as well as Pix2Pix versions), used their backbone networks to extract the image

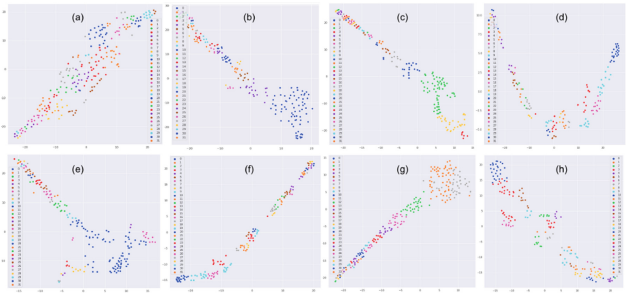

Fig. 7. Clustering partial NORCE-PV dataset image representations produced by (a) ResNet18 ImageNet weights (b) SimSiam random aug. (c) SimSiam random + PixPix based SGD aug. (d) SimCLR random + SGD aug. (e) SimSiam random + SGD aug. f) SimSiam with PixPix based SGD aug. (g) SwAV with SGD aug. (h) SimSiam with SGD aug.

representations (vectors) of 20% of the dataset, and then used KMeans and t-SNE methods to cluster and visualize the clusters. The outcome is presented in Fig. 7. Evidently, SGD, when pretrained on unlabelled images, enables better image representations (vectors, embeddings) in all of the tested cases. The more distant the clusters, the better the representations generated by the SSL backbone network. In fact, observing this clear delineation between the clusters on a 2D space had originally given us an idea of applying SGD as an augmentation on larger scales with other combinations on a downstream image segmentation task.

3.4.2 Image Segmentation and Object Detection Using SSL Backbones

As a downstream image segmentation architecture, we used Detectron2 [40] framework on MultiRes-PV dataset to detect and segment solar panels from rooftop aerial images. In order to pretrain the SSL algorithms, the following augmentation policies are applied (p within the parenthesis corresponds to the probability of applying the respective technique, and sample augmentations are shown in Fig. 8):

- Default SSL augmentations (*RandomResizedCrop, RandomRotate, RandomHorizontalFlip, RandomVerticalFlip, RandomColorJittering, RandomGrayscale, GaussianBlur at various probabilities, p*)
- SGD ($p = 1.0$),
- SGD ($p = 1.0$) with default SSL augmentations
- SGD ($p = 0.5$) with default SSL augmentations
- SGD ($p = 1.0$) with jittering ($p = 1.0$)
- SGD ($p = 0.5$) with jittering ($p = 1.0$)
- SGD ($p = 0.8$) with jittering ($p = 0.8$)

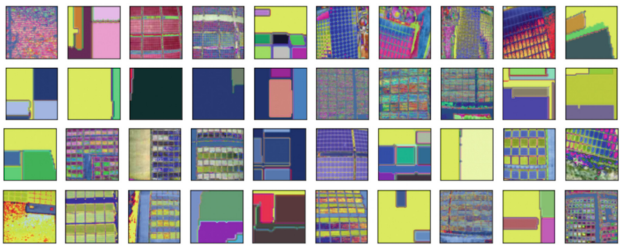

Fig. 8. This figure shows a batch of 40 NORCE-PV images. Every image in this batch is segmented by SGD at first and then the other default random augmentations are applied. Basically, this is what the model sees at each iteration.

The batch size is known to be one of the most important parameters in SSL and the larger the batch size, the better the representations as most of SSL

algorithms greatly benefit from large batches (in order to have larger number of negatives available in mini batches) but we could not investigate performance for more than 128 images in a batch due to limited computational resources and the high number of experiments that we need to run (more than 500 runs, each takes at least a few hours). In order to test the impact of image resizing and batch sizing on SGD in image segmentation tasks, we picked three parameter combinations with five SSL methods (SimSiam, SimCLR, BYOL, MoCo, Barlow Twins): (i) Offline SGD applied on the original size ($100p$) with batch size 16 ($bs16$), (ii) Offline SGD applied on the original size ($100p$) with batch size 128, (iii) Offline SGD applied on the upscaled size ($200p$) with batch size 16.

Upscaling (x2) is tested to see the impact of detailed SGD segmentations, and it is found that the performance reaches its peak when SGD is applied on the original image-size with a batch size of 128. We also tested with doubling the resolution of each image and then applying SGD but the impact of SGD on SSL performance was not as good as in the original image-size. This can be explained by the fact that the color variation in SGD with low resolution (original size) images is more or less similar to what we can expect to see in a solar panel segmentation. Once we increase the resolution by up-scaling, the details on a solar panel become more evident and the SGD may assign different colors for such areas, which we do not want to see for a simple panel segmentation task (e.g., every object is represented by a single color in image segmentation).

Then, using 500 images as a training set (out of 645 images from MultiRes-PV dataset), we pretrained all of the five SSL methods in a training loop with image segmentation and object detection tasks and measured the test accuracy on the test set (145 images). We observed that performances of SSL methods vary given the augmentation policy, as shown in Fig. 9. For instance, while MoCo performs better compared to other SSL methods, with default SSL augmentations yet no SGD, SimSiam and BYOL perform better compared to other SSL methods with SGD plus default SSL augmentations as well as with only default augmentations. In another case, SimCLR performs the best when SGD is applied at $p = 0.5$ along with default augmentations. This result tells us that each SSL method performs differently, as a function of the augmentation policy, while the impact of SGD also varies under different settings.

Then, we experimented with the same SSL methods, having additionally no-SSL (with default ImageNet checkpoints) at various levels of labeled images (subsets from 10% to 100%) and measured the test accuracy on the same 145 images. For instance, we picked at first 50 labeled images from the training set, then using a pretrained SSL backbone from the previous step, we trained an image segmentation and object detection model with Detectron2, and tested the models on the test set. Then we did the same for 20%, 30% and so on. The AP metrics for the selected SSL methods for selected augmentation are shown in Fig. 10. As seen from this figure, in every level of the reduced training set, an augmentation policy with an SGD component usually performs better than an SSL with default augmentations as well as a vanilla (no-SSL) model training. In some cases, using only SGD augmentation would even suffice without further application of augmentation.

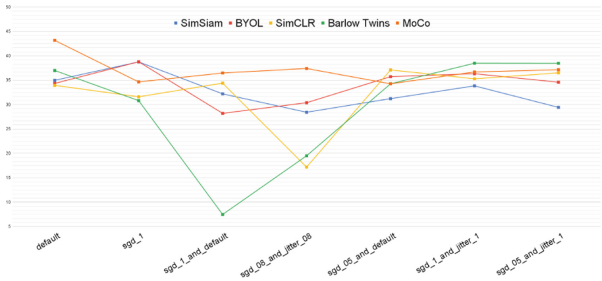

Fig. 9. Selecting the best combination of SGD and default augmentation strategies for various SSL methods using original size images with batch size 128. The performance of augmentation strategies varies by the SSL methods accompanied. SGD alone without any other augmentation strategies produced the best result in BYOL and SimSiam compared to other methods and augmentation policies.

3.4.3 Image Classification Using SSL Backbones

As a downstream image classification model, using CIFAR10 and STL10 datasets, we used Logistic Regression to classify the image representations taken from the penultimate layers of SimCLR backbones that are pretrained on CIFAR10 training set (5000 unlabelled images) with the augmentation policies mentioned before (with hashing applied to SGD augmentations). During these experiments, various levels of labeled images from CIFAR10 (from 10% to 100%) are picked at each iteration and only one SSL method (SimCLR) is picked for the sake of brevity. The results from this experiment can be seen in Fig. 11). Even though there is a slim performance gain (around 5% improvement) with SGD plus default augmentations applied, the default pretrained ImageNet backbone still performs better without SSL. This can be explained by the smaller number of utilized epochs (100) while pretraining with SimCLR and the large number of labelled images being used in the ImageNet pretraining. Considering that SimCLR is pretrained only by 5000 unlabelled images from CIFAR10 while the default ResNet18 model is pretrained with the entire ImageNet dataset with ground truth labels, getting similar metrics with SimCLR is still worth mentioning herein.

We run a similar experiment on the STL10 dataset (using 100 thousand unlabeled images), but SGD did not contribute in that case and performed worse. Since STL is already a subset of the ImageNet dataset, the default pretrained ImageNet backbone did quite well and exceed all the metrics achieved with SSL methods. The results are shown in Fig. 12. Given the fact that the images in the STL10 dataset have higher resolution (96×96) than the images in CIFAR10 (32×32), getting worse performance with SSL using SGD can be explained by the same phenomenon we observed in SGD of upscaled images doing worse compared to original size images.

4 Discussion

Our experiments with clustering the image representations via SGD-augmented backbone networks indicate that SGD enables obtaining better image representations in all of the cases that we tested. Even if we may not know the number of clusters within a dataset, the clear delineation between the clusters indicates a better representation to separate one image from another using visual clues. Usually, the most salient details and objects play the role of separators between one image from another, but using classical image augmentation policies (e.g., cropping, jittering, etc.) totally fails on this end. At the same time, SGD-based salient object segmentation does a better job on catching those critical details, thus assisting more in downstream clustering tasks.

We argue that using a salient image segmentation in SSL generates better representations when the downstream task is image segmentation or object detection due to the fact that salient object detection and segmentation approaches segment only the salient foreground object from the background, rather than partition an image into regions of coherent properties as in general segmentation algorithms. Our experiments with various augmentation combinations with and without SGD show that the augmentation policy having an SGD component usually performs better than an SSL with default augmentations as well as a vanilla (no SSL) model training. In some cases, using only SGD augmentation would even suffice, with no further application of augmentation.

Another important observation is that each SSL method performs differently as a function of the underlying augmentation policy, whereas the impact of SGD also varies under different settings. For instance, while MoCo performs better compared to other SSL methods, with default SSL augmentations yet no SGD, SimSiam and BYOL perform better compared to other SSL methods with SGD plus default SSL augmentations as well as with only default augmentations, as was elaborated in the previous section. Notably, this in fact the common question in the age of DL in which there are a myriad of alternatives: How to know which algorithm does better on a certain use case and how to pick the right one? Not only the SSL method but also the augmentation technique, downstream task and even the resolution of your images matter.

One of the most unexpected observation of our tests can be regarded as having worse results with SGD applied on high resolution images. We have observed this effect on various occasions during our experiments as reported in the previous section. This can be explained by the fact that the color variation in SGD with low resolution (original size) images is lower and more similar to what we can expect to see in a coarse grained segmentation tasks. Once we increase the resolution by up-scaling, the details on an image becomes more evident and SGD may assign different colors for such areas, which we do not want to see for a simple image segmentation task (e.g., every object is represented by a single color in image segmentation). However, SSL performing well on low resolution images can also be attributed to the low number of epochs (i.e., 100) as the model is not able to catch the low level details in high resolution images in such a limited number of epochs.

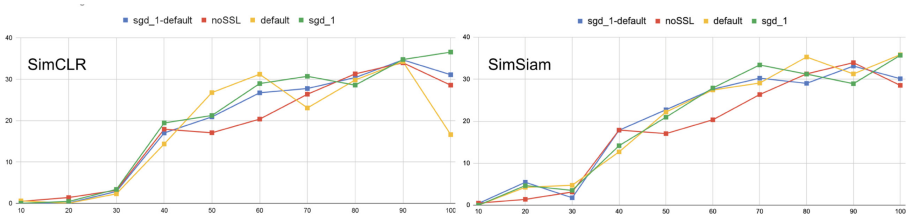

Fig. 10. Segmentation metrics for SGD with SimSiam and SimCLR experiments with reduced number of images from the training set at various levels. x-axis denotes the percentage of the samples with respect to entire training set (from 10% to 100%) and y-axis denotes test set average precision (AP) of segmentation model trained with the partial training set.

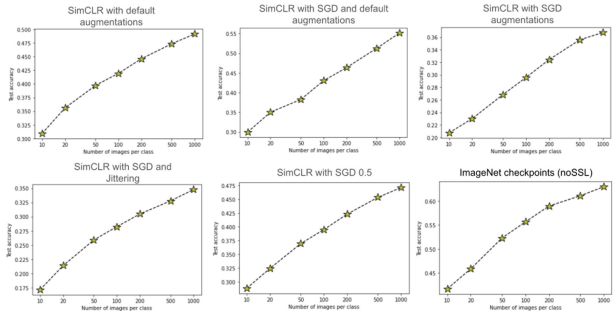

Fig. 11. Classification of images in CIFAR10 dataset using partial samples from training set and SSL method with SGD applied. SGD plus default augmentations outperforms default SSL augmentations by 5% and a little worse than ImageNet checkpoints.

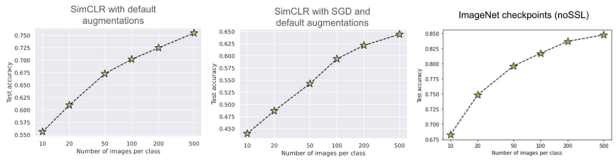

Fig. 12. Classification of images in STL10 dataset using partial samples from training set and SSL method with SGD applied. SGD did worse than default SSL augmentations and the default pretrained ImageNet backbone outperforms all the metrics achieved with SSL methods (due to the fact that STL is already a subset of ImageNet dataset).

Nevertheless, a recent SSL study also supports our claim and sheds some light on this issue as follows: [41] claims that current self-supervised methods learn representations that can easily disambiguate coarse-grained visual concepts like those in ImageNet. However, as the granularity of the concepts becomes finer, self-supervised performance lags further behind supervised baselines.

5 Conclusions

By empirically addressing the posed research questions, our investigation confirmed the capacity of SGD to play a role in modern SSL. Overall, this study successfully leveraged SGD, a 10-years-old salient object detection and segmentation algorithm, as a potent image augmentation technique for downstream image segmentation tasks. To achieve an effective integration of SGD into SSL pretraining routines, we devised a simple manipulation called offline augmentation with hashing. This fine implementation detail enabled us to run hundreds of SSL experiments with various parameters and configurations. We then demonstrated that using a salient image segmentation in SSL generates better representations when the downstream task is image segmentation. Our experiments with clustering the image representations via SGD-augmented backbone networks indicate that SGD helps better image representations in all of the cases tested. Our experiments with various augmentation policies including SGD show that the augmentation policy having a SGD component usually does better than a SSL with default augmentations; in some cases, using only SGD augmentation alone would even be better.

Our experiments with MultiRes-PV, CIFAR10 and STL10 also showed that SSL with SGD-based augmentation policy performs well with low resolution images. This still remains to be verified whether fine-grained features and/or the low number of epochs played a role in this observation. We contend that salient object segmentation algorithms produce coarse grained segmentation due to saliency, thus perform well on segmenting coarse grained objects like PV solar panels and it may not hold true if our goal were to extract fine grained details in an image.

We observed that each SSL method performs differently given the augmentation policy, whereas the impact of SGD also varies under different settings. In our opinion, the most unexpected observation of our tests can be regarded as having worse results with SGD applied to high resolution images compared to low resolution ones. We conclude that the augmentation technique, type of a downstream task and image resolution are the most important elements that have a high impact on the success of a SSL method picked.

As a future work, harnessing the offline augmentation with hashing, we plan to investigate if the observations gathered during this study would still hold true with any other unsupervised or zero-shot segmentation as well as salient object detection algorithms that are used as an augmentation policy in SSL pretraining process.

References

1. Leordeanu, M.: Unsupervised Learning in Space and Time. ACVPR, Springer, Cham (2020). https://doi.org/10.1007/978-3-030-42128-1

2. Chen, Y., Schmid, C., Sminchisescu, C.: Self-supervised learning with geometric constraints in monocular video: connecting flow, depth, and camera. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 7063–7072 (2019)
3. Jaiswal, A., Babu, A.R., Zadeh, M.Z., Banerjee, D., Makedon, F.: A survey on contrastive self-supervised learning. Technologies **9**(1), 2 (2020)
4. Ciga, O., Martel, A.L.: Learning to segment images with classification labels. Med. Image Anal. **68**, 101912 (2021)
5. Liu, X., et al.: Generative or contrastive. IEEE Trans. Knowl. Data Eng. Self-supervised Learn. (2021)
6. Cheng, M., Zhang, G., Niloy J, et al.: Global contrast based salient region detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Colorado Spring, USA, pp. 409–416 (2011)
7. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 649–666. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_40
8. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. ACM Trans. Graph. (ToG) **35**(4), 1–11 (2016)
9. Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 577–593. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_35
10. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: feature learning by inpainting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2536–2544 (2016)
11. Caron, M., et al.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9650–9660 (2021)
12. Lee, D.-H., et al.: Pseudo-label: the simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on Challenges in Representation Learning, ICML, vol. 3, p. 896 (2013)
13. Xie, Q., Luong, M.-T., Hovy, E., Le, Q.V.: Self-training with noisy student improves imagenet classification. arxiv e-prints, art. arXiv preprint arXiv:1911.04252 (2019)
14. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 69–84. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_5
15. Ledig, C., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4681–4690 (2017)
16. Xu, D., Xiao, J., Zhao, Z., Shao, J., Xie, D., Zhuang, Y.: Self-supervised spatiotemporal learning via video clip order prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10334–10343 (2019)
17. Afouras, T., Asano, Y.M., Fagan, F., Vedaldi, A., Metze, F.: Self-supervised object detection from audio-visual correspondence. arXiv preprintarXiv:2104.06401 (2021)
18. Dhere, A., Sivaswamy, J.: Self-supervised learning for segmentation. arXiv preprintarXiv:2101.05456 (2021)

19. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. Adv. Neural Inf. Process. Syst. **33**, 9912–9924 (2020)
20. 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, pp. 9729–9738 (2020)
21. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprintarXiv:2003.04297 (2020a)
22. Grill, J.-B., et al.: Bootstrap your own latent-a new approach to self-supervised learning. Adv. Neural Inf. Process. Syst. **33**, 21271–21284 (2020)
23. Nakamura, H., Okada, M., Taniguchi, T.: Self-supervised representation learning as multimodal variational inference. arXiv preprintarXiv:2203.11437 (2022)
24. Chen, T., Kornblith, S., Swersky, K., Norouzi, M., Hinton, G.E.: Big self-supervised models are strong semi-supervised learners. Adv. Neural Inf. Process. Syst. **33**, 22243–22255 (2020b)
25. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607. PMLR (2020c)
26. Kolesnikov, A., Zhai, X., Beyer, L.: Revisiting self-supervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1920–1929 (2019)
27. Balestriero, R., Bottou, L., LeCun, Y.: The effects of regularization and data augmentation are class dependent. arXiv preprintarXiv:2204.03632 (2022)
28. Rother, C., Kolmogorov, V., Blake, A.: "grabcut" interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. (TOG) **23**(3), 309–314 (2004)
29. Everingham, M., Gool, L.V., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
30. Ji, X., Henriques, J.F., Vedaldi, A.: Invariant information clustering for unsupervised image classification and segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9865–9874 (2019)
31. Kanezaki, A.: Unsupervised image segmentation by backpropagation. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1543–1547. IEEE (2018)
32. Kim, W., Kanezaki, A., Tanaka, M.: Unsupervised learning of image segmentation based on differentiable feature clustering. IEEE Trans. Image Process. **29**, 8055–8068 (2020)
33. Jiang, H., et al.: Multi-resolution dataset for photovoltaic panel segmentation from satellite and aerial imagery. Earth Syst. Sci. Data **13**(11), 5389–5401 (2021)
34. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
35. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 215–223. JMLR Workshop and Conference Proceedings (2011)
36. Hamilton, M., Zhang, Z., Hariharan, B., Snavely, N., Freeman, W.T.: Unsupervised semantic segmentation by distilling feature correspondences. arXiv preprint arXiv:2203.08414 (2022)
37. Zhou, X., Girdhar, R., Joulin, A., Krähenbühl, P., Misra, I.: Detecting twenty-thousand classes using image-level supervision. arXiv preprint arXiv:2201.02605 (2022)

38. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1125–1134 (2017)
39. Zbontar, J., Jing, L., Misra, I., LeCun, Y., Deny., S.: Barlow twins: self-supervised learning via redundancy reduction. In: International Conference on Machine Learning, pp. 12310–12320. PMLR (2021)
40. Wu, Y., Kirillov, A., Massa, F., Lo, W-Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)
41. Cole, E., Yang, X., Wilber, K., Mac Aodha, O., Belongie, S.: When does contrastive visual representation learning work? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14755–14764 (2022)