

Chapter

# Self-Supervised Contrastive Representation Learning in Computer Vision

*Yalin Bastanlar and Semih Orhan*

## Abstract

Although its origins date a few decades back, contrastive learning has recently gained popularity due to its achievements in self-supervised learning, especially in computer vision. Supervised learning usually requires a decent amount of labeled data, which is not easy to obtain for many applications. With self-supervised learning, we can use inexpensive unlabeled data and achieve a training on a pretext task. Such a training helps us to learn powerful representations. In most cases, for a downstream task, self-supervised training is fine-tuned with the available amount of labeled data. In this study, we review common pretext and downstream tasks in computer vision and we present the latest self-supervised contrastive learning techniques, which are implemented as Siamese neural networks. Lastly, we present a case study where self-supervised contrastive learning was applied to learn representations of semantic masks of images. Performance was evaluated on an image retrieval task and results reveal that, in accordance with the findings in the literature, fine-tuning the self-supervised training showed the best performance.

**Keywords:** self-supervised learning, contrastive learning, representation learning, computer vision, deep learning, pattern recognition

## 1. Introduction

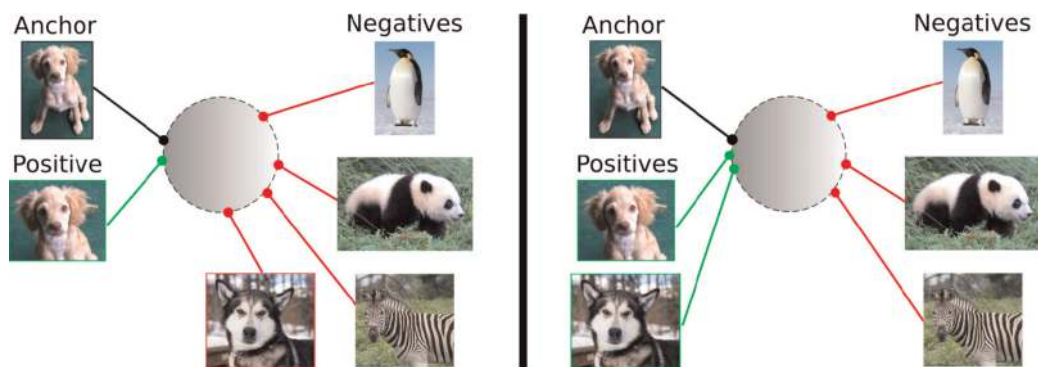
For an effective training, supervised learning requires a decent amount of labeled data, which is expensive. Unlabeled and inexpensive data (e.g. text and images on the Internet) is considerably more than the limited size datasets labeled by humans. We can use unlabeled data and perform a training on a pretext task, which is a **self-supervised** approach since we do not use the labels in our real task. Although the task and the defined loss are not the ones in our actual objective, we can still learn some representations that are valuable enough to be used for the final task. We basically learn a parametric mapping from the input data to a feature vector or tensor. In most cases, a smaller amount of labeled data is used to fine-tune the self-supervised training.

Although its origins date as back as 1990s [1, 2], contrastive learning has recently gained popularity due to its achievements in self-supervised learning, especially in computer vision. In **contrastive learning**, a representation is learned by comparing among the input samples. The comparison can be based on the similarity between positive pairs or dissimilarity of negative pairs. The goal is to learn such an embedding space in which similar samples stay close to each other while dissimilar ones are far apart. Contrastive learning can be applied to both supervised and unsupervised settings. Let us consider image classification problem. In supervised setting, positive pairs are different instances with the same label and negative samples are selected from other labels (**Figure 1**). On the other hand, in unsupervised (or self-supervised) setting, positive pairs are parts (or augmented versions) of the same instance and negative samples are other instances with any label. Khosla *et al.* [3] provide a performance comparison between supervised and self-supervised training for image classification problem. Also, a more comprehensive review of contrastive learning can be found in [4].

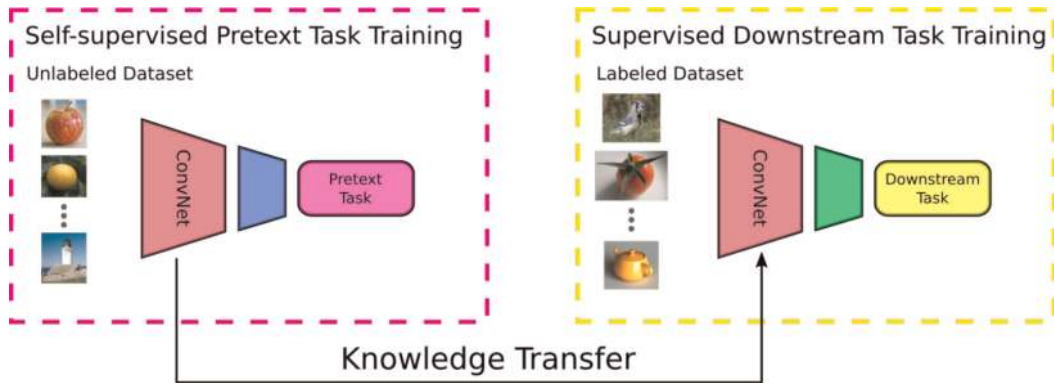
Since a self-supervised model does not know the actual labels corresponding to the inputs, its success depends on the design of the pretext tasks to generate the pseudo-labels from part of the input data. With these pseudo-labels, training on pretext task is performed with a ‘supervised’ loss function. Final performance on the pretext task is not important, but we hope that the learned intermediate representations can capture good information and be beneficial to a variety of downstream tasks.

Especially in computer vision and natural language processing (NLP), deep learning has become the most popular machine learning approach [5]. In parallel, self-supervised learning studies in computer vision have employed CNNs. **Figure 2** shows the knowledge transfer from a self-supervised training to a supervised one in a deep learning setting. We save convolutional layers which are assumed to produce learned representations. We change/add fully connected layers, place a classifier head and train with the limited amount of labeled data for a downstream task like image classification or object detection.

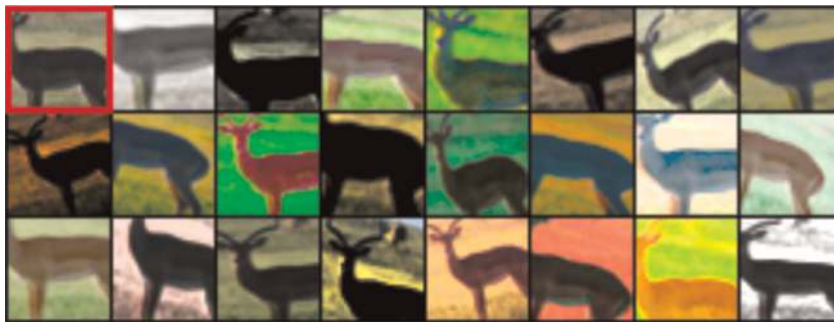
The remainder of this chapter is structured as follows. Pretext tasks that are common in literature are reviewed in Section 2. Section 3 has detailed information about recent self-supervised learning models that use Siamese architectures. Section 4 provides our own experimental study where self-supervised contrastive learning is employed to learn representations of semantic segmentation masks, which is followed by the conclusions in Section 5.



**Figure 1.** Self-supervised (left) vs. supervised (right) contrastive learning. Training results in an embedding space such that similar sample pairs stay close to each other while dissimilar ones are far apart. Figure is reproduced based on [3].



**Figure 2.** A model is first trained with a pretext task with unlabeled data, then fine-tuned on the downstream task with limited amount of labeled data. Usually convolution layers, which are mostly responsible of learning representations, are transferred. A few fully-connected layers towards the end are changed or retrained.



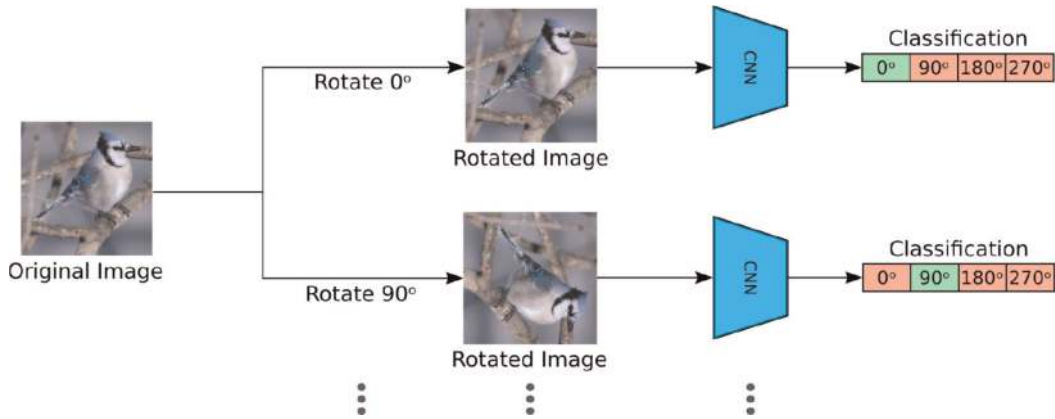
**Figure 3.** Several random transformations applied to a patch from the unlabeled dataset to be used for self-supervised learning. Original sample is in top-left. The idea was first used by [6] and the figure is from the original paper with author's permission.

## 2. Pretext tasks for self-supervised learning

**Image Distortion.** We expect that when an image goes through a small amount of distortion, its semantic meaning does not change. Dosovitskiy *et al.* [6] used this idea to create an exemplar-based classification task, where a surrogate class is formed with each dataset sample by applying a variety of transformations, namely translation, scaling, rotation, contrast and color (**Figure 3**). When this approach is applied to whole image instances, it can be called as ‘instance discrimination’ [7], where augmented versions of the same image (positive pair) should have similar representations and augmented versions of the different images (negative pair) should have different representations.

This is not only one of the first pretext tasks but also a very popular one. We will see in Section 3 that the mentioned type of augmentations have succeeded in learning useful representations and have achieved state-of-the-art results in transfer learning for downstream computer vision tasks.

**Image Rotation.** Each input image is first rotated by a multiple of  $90^\circ$  at random. A model is trained to predict the amount of rotation applied [8]. In basic setting, it is a 4-class classification problem, but different versions can be conceived. To estimate the amount of rotation, this pretext task forces the model learn semantic parts of objects,



**Figure 4.** Self-supervised representation learning by rotating input images, implemented in [8]. The model classifies the rotation [0°, 90°, 180°, 270°].

such as arms, legs, eyes. Thus, it would serve well for a downstream task like object recognition (Figure 4).

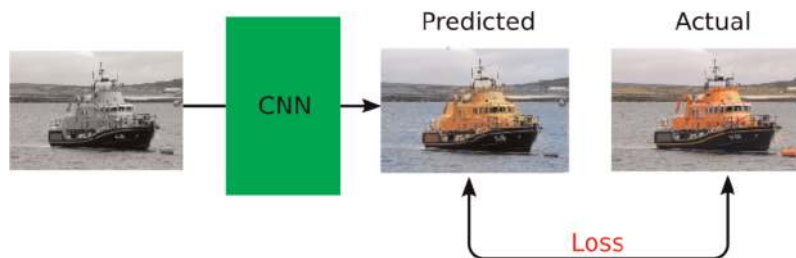
**Jig-saw Puzzle.** Noroozi and Favaro [9] designed a jigsaw puzzle game, where a CNN model is trained to place 9 shuffled patches back to the original locations. Each patch is processed independently with shared weights and a probability vector estimated per patch. Then, these estimations were merged to output a permutation.

**Image Colorization.** The task is to colorize gray-scale images into colorful images [10]. A CNN is trained to predict the colorized version of the input (Figure 5). Obtaining a training dataset is inexpensive since training pairs can be easily generated. Model’s latent variables represent grayscale images and can be useful for a variety of downstream tasks.

**Image Inpainting.** The pretext task is filling in a missing piece in the image (e.g. Pathak *et al.* [11]). The model is trained with a combination of the reconstruction (L2) loss and the adversarial loss. It has an encoder-decoder architecture and encoder part can be considered as representation learning.

Last two pretext tasks (image colorization and inpainting) and some other GANs (e.g. image super-resolution [12]) are generation-based methods, where a missing info in the content is generated from available input. Whereas distortion, rotation and jigsaw are context-based self-supervision methods. For more detailed literature on pretext tasks, we refer the readers to the review in [13].

In our study, we concentrate on the context-based approach. Taking advantage of contrastive learning, this approach nowadays achieves state-of-the-art performance



**Figure 5.** A model is trained to predict the colorized version of grayscale images (obtaining the dataset is inexpensive).

[14–17]. We will go into details, especially the models with Siamese architecture, in Section 3. The generation-based and context-based method distinction also exists for video representation learning. In [18], an encoder network is used to learn video representations. Then, a decoder uses the representations to predict future frames. Differently, Qian *et al.* [19] employ contrastive learning with distortions (augmentations) to learn representations and to classify video clips.

The rest of our chapter will consider works on image data. Before proceeding, let us give a few examples where contrastive learning is used for image-text pairs. Contrastive Language-Image Pre-training (CLIP, [20]) is a pretext task, where a text encoder and an image encoder are jointly trained to match captions with images. Training set consists of 400 million (image,text) pairs and an inter-modal contrastive loss is defined such that image and text embeddings of same objects will be closer to each other. Then, this pretraining is employed for a downstream task of zero-shot class prediction from images. Li *et al.* [21] performed a similar task for semantic segmentation. An image encoder is trained with a contrastive objective to match pixel semantic embeddings to the text embeddings. Another example presents contrastive learning of medical visual representations from paired images and text [22].

### 3. Self-supervised contrastive learning models

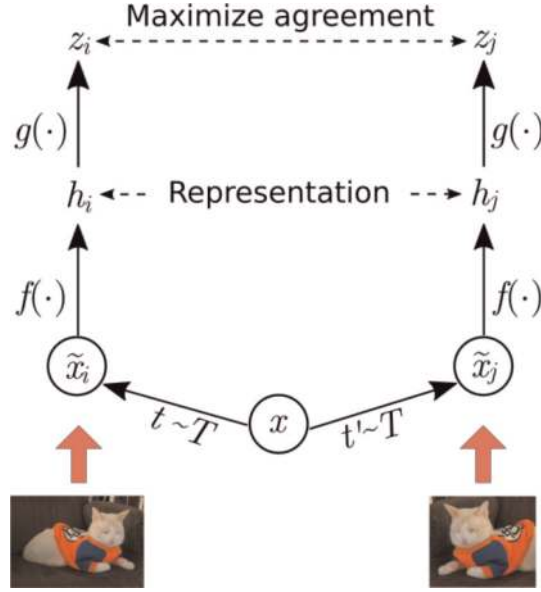
The goal of contrastive learning is to learn such an embedding space in which similar sample pairs stay close to each other while dissimilar ones are far apart. Implemented using Siamese networks, recent approaches create two different augmentations of samples and feed into the networks for contrastive learning. While SimCLR [14] and MoCo [15] use the negative samples directly along with the positive ones, BYOL [16] and SimSiam [17] achieved similar performance just with the positive samples. Differently, SwAV [23] forced consistency between cluster assignments of augmentations, instead of comparing features directly. Shortly after, vision transformers were included in self-supervised learning architectures [24, 25]. According to the results, not only image classification, but also object detection and semantic segmentation as downstream tasks benefit from self-supervised contrastive learning. Let us briefly explain some of these main approaches.

#### 3.1 SimCLR

Let us describe SimCLR [14] first, then we will describe other methods by comparing to previous ones. SimCLR uses both positive and negative samples, but being positive or negative does not correspond to actual class labels. Augmented versions of the anchor are taken as positives, whereas samples belong to different instances are taken as negatives (**Figure 6**).

- Let  $T$  be the set of image transformation operations where  $t \sim T$  and  $t' \sim T$  are two different transformation operators independently sampled from  $x$ . These transformations are random cropping and resizing, random Gaussian blur and random color distortion. A  $(\tilde{x}_i, \tilde{x}_j)$  pair of query and key views is positive when these two views are created by applying different transformations on the same image  $x$ :  $\tilde{x}_i = t(x)$  and  $\tilde{x}_j = t'(x)$ .




**Figure 6.**

SimCLR framework [14], where two separate data augmentations are sampled from a predefined family of augmentations (crop, blur color jitter). An encoder network  $f(\cdot)$  and a projection head  $g(\cdot)$  are trained to maximize the agreement between the embeddings of these two samples. When the self supervised learning is over, projection head can be thrown away.

- A base feature encoder  $f(\cdot)$  then extracts the representations from all the augmented data samples  $h_i = f(\tilde{x}_i)$ ,  $h_j = f(\tilde{x}_j)$ . There is no restriction on the choice of the encoder's architecture but a ResNet-50 [26] model was preferred for SimCLR due to its simplicity. The representation  $h$  in this case is the output of the average pooling layer of ResNet-50.
- Each representation  $h$  is then fed into a projection head  $g(\cdot)$  to map representations to the embedding space where the contrastive loss is applied.  $z_i = g(h_i)$ ,  $z_j = g(h_j)$ . This projection head can be as simple as a one-layer multi-layer perceptron (MLP) using a non-linear activation.
- A batch of  $(z_i, z_j)$  pairs representing the embeddings from two augmented versions of the same image, are then fed into the contrastive loss function which encourages the distance between embeddings from positive pairs to be small and the distances of embeddings from negative pairs to be large.

SimCLR uses the contrastive loss given in Eq. (1). This is a categorical cross-entropy loss to identify the positive sample among a set of negative samples (inspired from InfoNCE [27]).

$$L^{self} = \sum_{i \in I} L_i^{self} = - \sum_{i \in I} \log \frac{\exp(z_i \cdot z_j / \tau)}{\sum_{a \in A(i)} \exp(z_i \cdot z_a / \tau)} \quad (1)$$

$N$  images are randomly taken from the dataset. Thus, the training batch consists of  $2N$  images to which data augmentations are randomly applied. Let  $i \in I \equiv \{1 \dots 2N\}$  be the index of an arbitrary augmented sample, then  $j$  is the index of the other

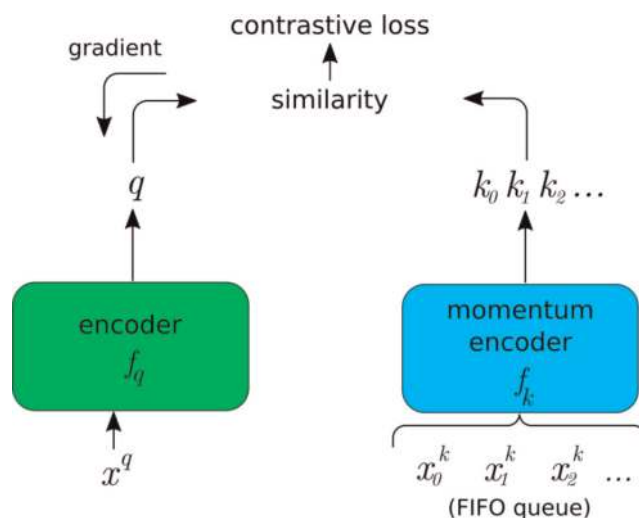
augmentation of the same original image.  $\tau \in R^+$  is a scalar temperature parameter,  $\cdot$  represents the dot product, and  $A(i) \equiv I - \{i\}$ . We call index  $i$  the anchor, index  $j$  is the positive, and the other  $2(N - 1)$  indices as negatives. The denominator has a total of  $2N - 1$  terms (one positive and  $2N - 2$  negatives).

A common protocol to evaluate self-supervised model efficiency is to place a linear classifier on top of (frozen) layers learnt by self-supervised training and train it for the downstream task with the labeled data. If the performance gap between this self-supervised encoder + linear classifier and a fully-supervised model is small, then the self-supervised training considered as efficient. An alternative evaluation protocol uses semi-supervised learning, i.e. pretrained network is re-trained as a whole with a certain percentage of available labels. Experiments reveal that re-training with only 10% of the labeled data achieves a performance (92.8%) very close to fully-supervised training performance on the whole dataset (94.2%) as reported in [14] (performances are top-5 classification accuracy on ImageNet dataset for ResNet-50).

### 3.2 Momentum contrast (MoCo)

Contrastive methods based on InfoNCE loss tend to work better with high number of negative examples since negative examples may represent underlying distribution more efficiently. SimCLR requires large batches (4096 samples) to ensure that there is enough negatives which demands high computation power (8 V100 GPUs in their study). To alleviate this need, MoCo [15] uses a dictionary of negative representations that is structured as a FIFO queue. This queue-based dictionary enables us to reuse representations of immediately preceding mini-batches of data. Thus, the main advantage of MoCo compared to SimCLR is that MoCo decouples the batch size from the number of negatives. SimCLR requires a large batch size and suffers performance drops when the batch size is reduced.

Given a query sample  $x^q$ , we get a query representation through our online encoder  $q = f_q(x^q)$ . A list of key representations  $\{k_0, k_1, k_2, \dots\}$  coming from the dictionary and are encoded by a different encoder  $k_i = f_k(x_i^k)$  as shown in **Figure 7**. Naming two



**Figure 7.** MoCo framework [15]. The encoder that takes negative samples (from a FIFO queue) is not updated by backpropagation but with the other encoder's parameters with a momentum coefficient. That's why it is called the momentum encoder.

compared representations as query and key is new, but one can think of them as the two augmentations of the same sample in SimCLR (**Figure 6**).

Let us assume that there is a single positive key,  $k_+$ , in the dictionary that matches  $q$ . Then, the contrastive loss with one positive and  $N - 1$  negative samples becomes:

$$L_{MoCo} = -\log \frac{\exp(q \cdot k_+/\tau)}{\sum_1^N \exp(q \cdot k_i/\tau)} \quad (2)$$

From the two encoders defined above, for  $f_k$  we can not apply backpropagation since it works on the queue. Copying online encoder's ( $f_q$ ) weights to  $f_k$  could be a solution, however MoCo proposed to use a momentum-based update with a momentum coefficient:

$$\theta_k \leftarrow m\theta_k + (1 - m)\theta_q \quad (3)$$

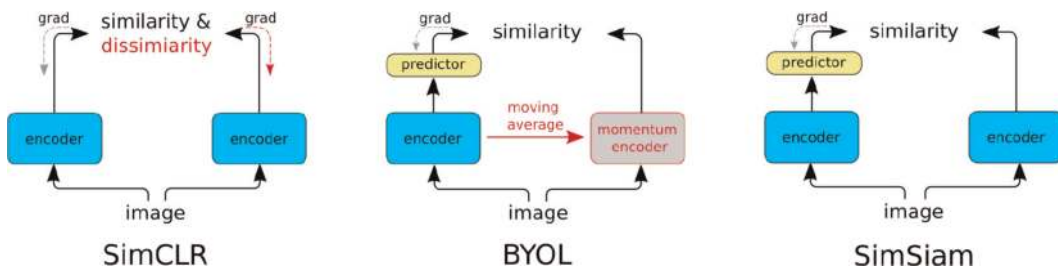
where  $m \in [0, 1]$  is the momentum coefficient and  $\theta_q$  and  $\theta_k$  are parameters of  $f_q$  and  $f_k$  respectively (**Figure 7**).

Later on, two design choices in SimCLR, namely MLP projection head and more stronger data augmentation were integrated into the approach resulting in MoCo-v2 [28].

### 3.3 Bootstrap your own latent (BYOL)

Different from the approaches above, BYOL [16] achieves similar representation performance without using negative samples. It relies on two different neural networks (in contrast to SimCLR but similar to MoCo), referred to as online and target networks that interact. Online network has a predictor head. Target network has the same network architecture with the online network except for the predictor head (**Figure 8**). Parameters of the target network are not updated with back-propagation, but with a moving average of online network's weights just as MoCo did for the momentum encoder.

It is curious that how the model escapes from collapsing (i.e. a trivial solution of fixed vector for each sample) when no negative samples are used. Authors of BYOL thought it is due to the momentum update, but later (with SimSiam [17]) it was discovered that using stop-gradient and predictor head is enough.



**Figure 8.** Comparison of some Siamese architectures: SimCLR, BYOL and SimSiam. Dashed lines indicate back-propagation. Components colored in red are no more needed in SimSiam. Figure is reproduced based on [17].



### 3.4 Simple Siamese (SimSiam)

BYOL needs to maintain two copies of weights for the two separate networks which can be resource demanding. SimSiam [17] solves this problem with parameter sharing between the networks (with and w/o predictor head). The encoder  $f(\cdot)$  shares weights while processing two views. A prediction MLP head, denoted as  $g(\cdot)$  transforms the output of only one view. Thus, two augmented views ( $x_1$  and  $x_2$ ) results in two outputs:  $p_1 = g(f(x_1))$  and  $z_2 = f(x_2)$ . Their negative cosine similarity is denoted as  $D(p_1, \text{stopgrad}(z_2))$ , where stopgrad operation is an important component. It implements that  $z_2$  is treated as a constant term and encoder receives no gradients from  $z_2$ . Gradient only flows back to the encoder through the prediction head.

Finally, negative cosine similarity based total loss is computed in a symmetric fashion:

$$L_{\text{SimSiam}} = \frac{1}{2}D(p_1, \text{stopgrad}(z_2)) + \frac{1}{2}D(p_2, \text{stopgrad}(z_1)) \quad (4)$$

**Figure 8** compares SimSiam with SimCLR and BYOL. SimSiam [17] does not use negative samples as SimCLR and MoCo did. Success with SimSiam also shows that momentum encoder (or any sort of moving average update of weights) is not needed. Stop-gradient operation and including predictor head are enough to prevent the model from collapsing.

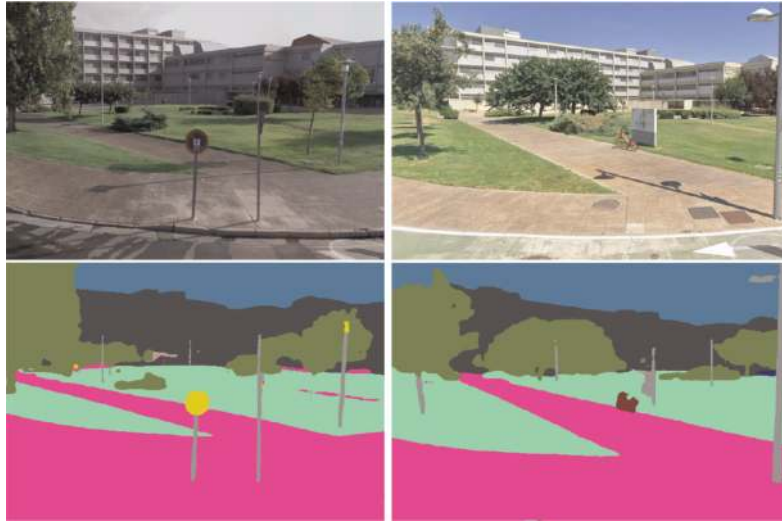
SimSiam also presents transfer learning results for object detection and semantic segmentation downstream tasks. Results reveal that starting with a self-supervised pre-training on ImageNet outperforms image classification pre-training on ImageNet.

### 3.5 Self-supervised vision transformers

Caron *et al.* [24] proposed another Siamese architecture where one of the network's parameters are updated with a moving average of other's parameters. More interestingly, they replaced encoder CNNs with vision transformers and reported increasing success for various downstream tasks. Shortly after, Li *et al.* [25] proposed a more efficient vision transformer architecture together with a new pre-training task which is based on region matching.

## 4. Case study: semantic mask representation learning

As a case study, we employ self-supervised contrastive learning to learn representations of semantically segmented images, i.e. semantic masks. This learning task is especially useful when two scenes are compared according to their semantic content. A use case would be image retrieval based localization, where standard approach extract features from RGB images and compare them to find the most similar image in the database [29, 30]. Recently, several studies showed that checking semantic resemblance between query and database images and using this similarity score while retrieving images improves localization accuracy [31–33]. The reason of improvement is that there is appearance difference between images taken at different times (query-database) due to illumination differences, viewpoint variations, seasonal changes.



**Figure 9.** The image on top-left was taken in 2008 and the image on top-right was taken in 2019 (source: Google street view) which respectively represent query and database for image retrieval. Observe illumination differences, viewpoint variations and changing objects. Bottom row shows their semantic segmentation results. Semantic similarity can help to verify/deny the localization result.

Although RGB image features are directly affected by those changes, semantic labels are stable most of the time (**Figure 9**).

Given a semantic mask, obtaining the most similar result among the alternatives is not a trivial task. SIFT-like features do not exist to match. Moreover, two masks of the same scene are far from being identical not only because of changing content but also due to camera position and viewpoint variations. Thus, instead of employing a pixel-by-pixel label comparison score, a trainable semantic feature extractor is preferable.

Measuring semantic similarity to distinguish if two images belong to the same scene or not is a task especially suitable for self-supervised learning. Because datasets has to be prepared such that query and database are the same scene but different images (preferably long-term difference) is not easy. However, large amount of semantic masks can easily be obtained for a self-supervised training. We do not need groundtruth masks, since a successful estimation is enough to compute semantic similarity.

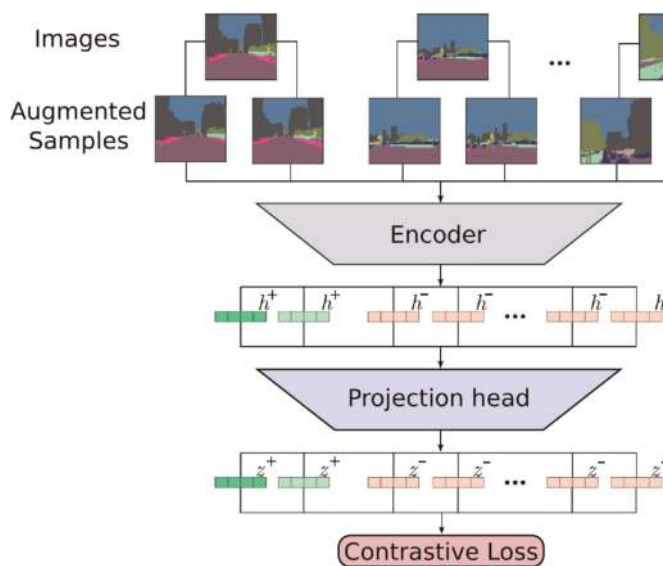
#### 4.1 Dataset and self-supervised contrastive training

Our unsupervised learning dataset composed of 3484 images randomly taken from UCF dataset [34]. These are perspective images obtained from Google Street View panoramas which where taken in Pittsburgh, PA before 2014. Our supervised training and test datasets have query-database image pairs. Query images were also taken from UCF dataset (not coinciding with the 3484 images mentioned above). Database images were collected again from Google Street View panoramas at the same locations of query images but in 2019. This time gap results in seasonal changes and illumination variances. Also, a wide camera baseline between the database and query images conforms better to the long-term localization scenario [35]. Top row in **Figure 9** shows an example of query-database image pair with time difference.

Since our aim to learn representations for semantic masks, we first automatically generated a semantic mask for each image in our dataset using a well-performing CNN model [36]. The CNN model we employed trained on Cityscapes [37], which is an urban scene understanding dataset consists of 30 visual classes. Examples are in **Figure 9** (bottom row). After this point, we only have semantic masks in our dataset.

We used SimCLR [14] as our contrastive learning model and trained a ResNet-18 as the encoder. Encoder network (**Figure 10**) produces  $h = Enc(x) \in R^{512}$  features, whereas projection network produces  $z = Proj(h) \in R^{512}$  features. We set batch size as 85 and resized semantic mask to  $64 \times 80$  resolution (due to GPU memory limitation) and used two different data augmentation methods during the training: random resized crop and random rotation. We set maximum rotation parameter as  $3^\circ$ , since severe rotations are not expected between query and database images. Crop parameter, however, is important to represent the variation in our dataset. Results for varying crop parameter values will be discussed in Section 4.2. Augmentation of semantic masks is visualized in **Figure 10**. Other augmentations (such as color jitter, horizontal flip, brightness and contrast distortions), which are common for image classification and object detection downstream tasks are not included since they are not expected distortions for semantic masks. We used AMSGrad optimizer which is a variant of Adam.

CNN model, trained as explained above, is now ready to produce a similarity score when two semantic masks (one query and one database) are given. After self-supervised training, same network can be fine-tuned with a labeled dataset (query and database segmentation masks for the same scene). For this purpose, we prepared a dataset of 368 query images with their corresponding database images and extracted their semantic masks. **Figure 9** shows an example of this preparation. Not surprisingly, this paired dataset is much smaller than the self-supervised training dataset. Here, common practice in literature is that the projection head (**Figure 10**) is removed



**Figure 10.**

*Illustration of training a CNN model with self-supervised contrastive loss on a dataset that consists of semantically segmented masks. A positive pair is created from two randomly augmented views of the same mask, while negative pairs are created from views of other masks. All masks are encoded by the a shared encoder and projection heads before the representations are evaluated by the contrastive loss function.*

after pretraining and a classifier head is added and trained with labeled data for the downstream task. However, our pretext and downstream tasks are the same. We learn semantic representations by treating each sample as its own class (exemplar-CNN [6], instance discrimination [7]). Thus, we do not place a classifier head, but we retrain the network (partially or full).

## 4.2 Experimental results

To be able to measure the capability of representing semantic masks, we conduct experiments that compare the retrieval accuracies of three training schemes. First is the CNN model which is trained with the supervised training set (368 query-database pairs). This is the baseline model that does not exploit self-supervised training at all. Second is the CNN model that is trained in a self-supervised fashion with 3484 individual semantic masks (no matching pairs). Lastly, the model with self-supervised training is retrained with the supervised training set. Two versions exist: *i*) only replacing dense layers and training them, *ii*) retraining all layers.

Trained models are tested on a test set which consists of 120 query-database pairs (different from 368 pairs used in training). Performances are compared with Recall@N metric. According to this metric, for a query image, the retrieval is considered successful if any of top-N retrieved database images is a correct match. In other words, Recall@1 is the recall when only the top-most retrieval is checked.

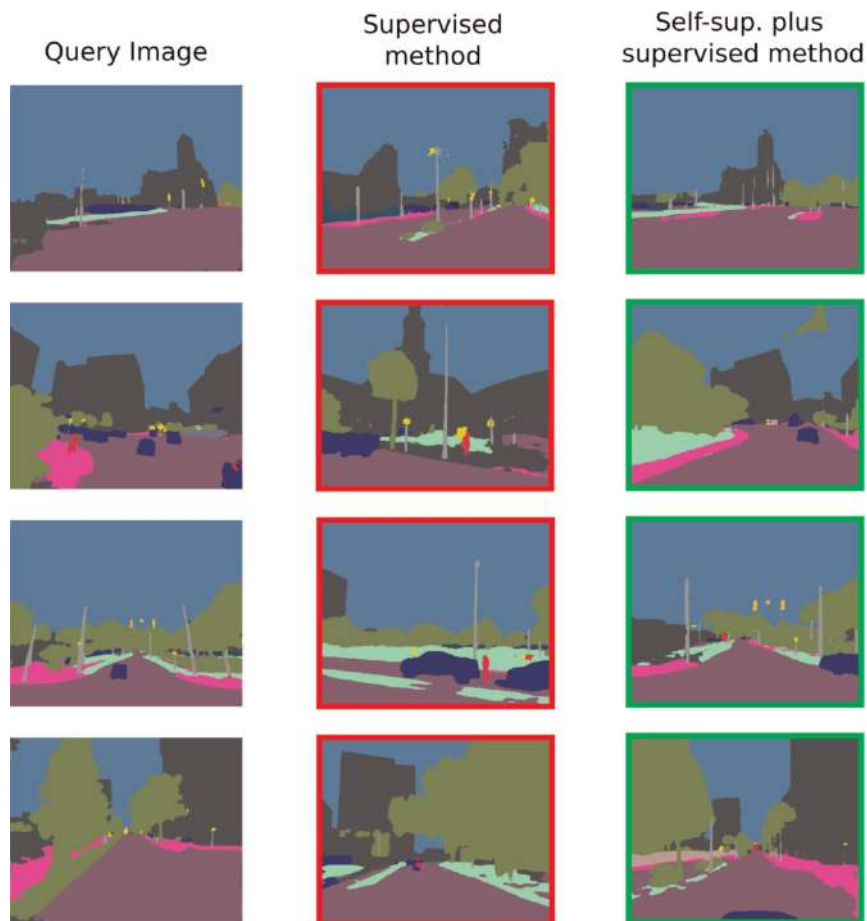
We observe in **Table 1** that, only supervised training is not very successful. In fact, for certain N values self-supervised training managed to outperform supervised training alone. This shows the power of self-supervised learning when a large dataset is provided. Our unlabeled dataset is much larger than the labeled dataset (3484  $\gg$  368). Regarding the two fine-tuning schemes, replacing dense layers and training them from scratch improved self-supervised training but not for all N values. On the other hand, fine-tuning all layers worked best by a considerable margin. Since our pretext and downstream tasks are the same (i.e. we do not train a classification head etc.), it is not surprising that replacing dense layers did not help much. **Figure 11** shows several examples where supervised training fails but the proposed self-supervised approach (after fine-tuning) succeeds.

**Table 2** presents the effect of minimum crop ratio parameter used in data augmentation module. Since it is an important parameter to represent the variation in our semantic masks, we compare the performance for minimum crop ratio from 0.9 to 0.1. Apart from individual Recall@N values, we also compute and plot mean recall (mean of all N values) in **Table 2** last column and in **Figure 12**. We observe that it is

Training methods	Retrieval accuracy (Recall@N)				
	N = 1	N = 2	N = 3	N = 4	N = 5
Only supervised training	0.500	0.608	0.767	0.808	0.817
Only self-supervised training	0.567	0.692	0.733	0.775	0.800
Dense layers were replaced and trained	0.542	0.675	0.767	0.825	0.850
All layers were fine-tuned	<b>0.633</b>	<b>0.758</b>	<b>0.808</b>	<b>0.858</b>	<b>0.867</b>

**Table 1.**

*Only supervised training is compared with self-supervised training and fine-tuned versions of it.*

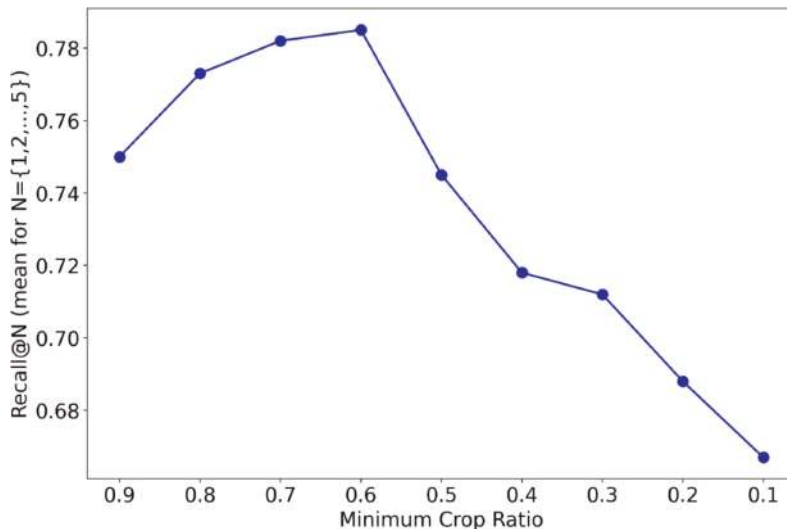


**Figure 11.** Each row shows a retrieval result for a given query (left column). Examples show the cases where only supervised training (middle column) fails at Recall@1, but utilizing self-supervised training and then fine-tuning on the labeled dataset (query-database pairs) correctly retrieves (last column).

Crop ratio	Retrieval accuracy (Recall@N)					
	N = 1	N = 2	N = 3	N = 4	N = 5	mean
0.90	0.608	0.708	0.758	0.817	0.858	0.750
0.80	0.617	0.733	0.800	0.848	0.867	0.773
0.70	0.617	0.742	<b>0.817</b>	<b>0.858</b>	<b>0.875</b>	0.782
0.60	<b>0.633</b>	<b>0.758</b>	0.808	<b>0.858</b>	0.867	<b>0.785</b>
0.50	0.617	0.700	0.767	0.808	0.833	0.745
0.40	0.575	0.692	0.717	0.783	0.825	0.718
0.30	0.567	0.675	0.742	0.767	0.808	0.712
0.20	0.542	0.633	0.717	0.767	0.783	0.688
0.10	0.525	0.608	0.675	0.742	0.783	0.667

**Table 2.** Effect of the minimum crop ratio parameter in data augmentation at the stage of retraining of the self-supervised model.





**Figure 12.** Mean Recall@N values for varying min. Crop ratio parameter. Observe the reverse U-shape with a peak at 0.6.

highest around 0.6 and 0.7. Performance gradually drops as we increase or decrease the minimum crop ratio. A minimum random crop parameter of 0.6 means that cropped mask covers at least 60% area of the original mask. Since query and database masks in our training and test datasets have a considerable overlap ratio, it is reasonable that 0.6 or higher overlaps serve best. This result is also in accordance with the finding in [38] that there is a reverse U-shape relationship between the performance and the mutual information within augmented views. When crops are close to each other (high mutual information, e.g. crop ratio = 0.9) the model does not benefit from them much. On the other hand, for low crop ratios (low mutual information) model can not learn well since views look quite different from each other. Peak performance stays somewhere in between.

## 5. Conclusions

In this chapter, we presented the main concepts in self-supervised contrastive learning and reviewed the approaches that attracted attention due to their success in computer vision. Contrastive learning that aims to end up in an embedding space where similar samples stay close to each other was implemented successfully with Siamese neural networks. Necessity on huge computation power was also alleviated with the most recent models. Currently, for common downstream tasks of computer vision such as object detection and semantic segmentation, self-supervised pre-training is a better alternative than using a model trained on ImageNet for image classification.

We also presented a case study where self-supervised contrastive learning is applied to learn representations of semantic masks of images. Performance was evaluated on an image retrieval task where the most similar semantic mask is retrieved from the database for a given query. In compliance with the results on other vision tasks in the literature, fine-tuning the self-supervised model with available labeled data gave better results than the supervised training alone.

## Acknowledgements

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under Grant No. 120E500 and also under 2214-A International Researcher Fellowship Programme.

## Abbreviations

CNN	Convolutional neural network
RGB	Red green blue
NLP	Natural language processing
LSTM	Long short term memory
GAN	Generative adversarial network
MoCo	Momentum contrast
BYOL	Bootstrap your own latent
MLP	Multi-layer perceptron
FIFO	First in first out

## Author details


Yalin Bastanlar\* and Semih Orhan

Department of Computer Engineering, Izmir Institute of Technology, Izmir, Turkey

\*Address all correspondence to: [yalinbastanlar@iyte.edu.tr](mailto:yalinbastanlar@iyte.edu.tr)

## IntechOpen

---

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

## References

- [1] Bromley J, Guyon I, LeCun Y, Sackinger E, Shah R. Signature verification using a siamese time delay neural network. *Processing Systems. Advances in Neural Information Processing Systems*. 1993:737-744
- [2] Becker S, Hinton GE. Self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*. 1992; 355:161-163
- [3] Khosla P, Teterwak P, Wang C, Sarna A, Tian Y, Isola P, et al. Supervised contrastive learning. *Advances in Neural Information Processing Systems*. 2020
- [4] Le-Khac PH, Healy G, Smeaton AF. Contrastive representation learning: A framework and review. *IEEE Access*. 2020
- [5] Tekir S, Bastanlar Y. Deep learning: Exemplar studies in natural language processing and computer vision. In: *Data Mining - Methods, Applications and Systems*. London: InTechOpen; 2020. DOI: 10.5772/intechopen.91813
- [6] Dosovitskiy A, Springenberg JT, Riedmiller M, Brox T. Discriminative unsupervised feature learning with convolutional neural networks. *Advances in Neural Information Processing Systems*. 2014
- [7] Wu Z, Xiong Y, Yu SX, Lin D. Unsupervised feature learning via non-parametric instance discrimination. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018
- [8] Gidaris S, Singh P, Komodakis N. Unsupervised representation learning by predicting image rotations. *ICLR*. 2018
- [9] Noroozi M, Favaro P. Unsupervised learning of visual representations by solving jigsaw puzzles. *European Conference on Computer Vision (ECCV)*. 2016
- [10] Zhang R, Isola P, Efros A. Colorful Image Colorization. *European Conference on Computer Vision (ECCV)*. 2016
- [11] Pathak D, Krahenbuhl P, Donahue J, Darrell T, Efros AA. Context encoders: Feature learning by inpainting. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016
- [12] Ledig C, Theis L, Huszar F, Caballero J, Cunningham A, Acosta A, 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 (CVPR)*. 2017
- [13] Jing L, Tian Y. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2021;43(11): 4037-4058. DOI: 10.1109/TPAMI.2020.2992393
- [14] Chen T, Kornblith S, Norouzi M, Hinton G. A simple framework for contrastive learning of visual representations. In: *International Conference on Machine Learning (ICML)*. 2020
- [15] He K, Fan H, Wu Y, Xie S, Girshick R. Momentum contrast for unsupervised visual representation learning. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020

- [16] Grill JB, Strub F, Alché F, Tallec C, Richemond PH, Buchatskaya E, et al. Bootstrap your own latent: A new approach to self-supervised learning. *Advances in Neural Information Processing Systems*. 2020
- [17] Chen X, He K. Exploring simple siamese representation learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021
- [18] Srivastava N, Mansimov E, Salakhutdinov R. Unsupervised learning of video representations using LSTMs. *International Conference on Machine Learning*. 2015
- [19] Qian R, Meng T, Gong B, Yang MH, Wang H, Belongie S, et al. Spatiotemporal contrastive video representation learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021
- [20] Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, et al. Learning transferable visual models from natural language supervision. *Proceedings of the 38th International Conference on Machine Learning*. 2021
- [21] Li B, Weinberger KQ, Belongie S, Koltun V, Ranftl R. Language-driven semantic segmentation. *International Conference on Learning Representations (ICLR)*. 2022
- [22] Zhang Y, Jiang H, Miura Y, Manning CD, Langlotz CP. Contrastive learning of medical visual representations from paired images and text. *arXiv:2010.00747v1*. 2020
- [23] Caron M, Misra I, Mairal J, Goyal P, Bojanowski P, Joulin A. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems (NeurIPS)*. 2020
- [24] Caron M, Touvron H, Misra I, Jegou H, Mairal J, Bojanowski P, et al. Emerging properties in self-supervised vision transformers. *International Conference on Computer Vision (ICCV)*. 2021
- [25] Li C, Yang J, Zhang P, Gao M, Xiao B, Dai X, et al. Efficient self-supervised vision transformers for representation learning. *International Conference on Learning Representations (ICLR)*. 2022
- [26] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016
- [27] van den Oord A, Li Y, Vinyals O. Representation learning with contrastive predictive coding. *arXiv:1807.03748*. 2018
- [28] Chen X, Fan H, Girshick R, He K. Improved baselines with momentum contrastive learning. *arXiv:2003.04297*. 2020
- [29] Arandjelovic R, Gronat P, Torii A, Pajdla T, Sivic J. Net VLAD: CNN architecture for weakly supervised place recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016
- [30] Ge Y, Wang H, Zhu F, Zhao R, Li H. Self-supervising fine-grained region similarities for large-scale image localization. *European Conference on Computer Vision (ECCV)*. 2020
- [31] Cinaroglu I, Bastanlar Y. Long-term image-based vehicle localization improved with learnt semantic descriptors. *Engineering Science and*

Technology, an International Journal. 2022;**35**:101098

[32] Cinaroglu I, Bastanlar Y. Training semantic descriptors for image-based localization. ECCV Workshop on Perception for Autonomous Driving. 2020

[33] Orhan S, Guerrero JJ, Bastanlar Y. Semantic pose verification for outdoor visual localization with self-supervised contrastive learning. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. 2022

[34] Zamir AR, Shah M. Image geo-localization based on multiple nearest neighbor feature matching using generalized graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2014;**36**(8):1546-1558

[35] Sattler T, Maddern W, Toft C, Torii A, Hammarstrand L, Stenborg E, et al. Benchmarking 6DOF outdoor visual localization in changing conditions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018

[36] Sun K, Zhao Y, Jiang B, Cheng T, Xiao B, Liu D, et al. High-resolution representations for labeling pixels and regions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2019

[37] Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, et al. The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016

[38] Tian Y, Sun C, Poole B, Krishnan D, Schmid C, Isola P. What makes for good views for contrastive learning? Advances in Neural Information Processing Systems (NeurIPS). 2020