# Fully Convolutional Network-Based Self-Supervised Learning for Semantic Segmentation

Zhengeng Yang, Hongshan Yu, Yong He, Wei Sun, Zhi-Hong Mao, *Senior Member, IEEE*, and Ajmal Mian, *Senior Member, IEEE*

***Abstract*—Although deep learning has achieved great success in many computer vision tasks, its performance relies on the availability of large datasets with densely annotated samples. Such datasets are difficult and expensive to obtain. In this article, we focus on the problem of learning representation from unlabeled data for semantic segmentation. Inspired by two patch-based methods, we develop a novel self-supervised learning framework by formulating the jigsaw puzzle problem as a patch-wise classification problem and solving it with a fully convolutional network. By learning to solve a jigsaw puzzle comprising 25 patches and transferring the learned features to semantic segmentation task, we achieve a 5.8% point improvement on the Cityscapes dataset over the baseline model initialized from random values. It is noted that we use only about 1/6 training images of Cityscapes in our experiment, which is designed to imitate the real cases where fully annotated images are usually limited to a small number. We also show that our self-supervised learning method can be applied to different datasets and models. In particular, we achieved competitive performance with the state-of-the-art methods on the PASCAL VOC2012 dataset using significantly fewer time costs on pretraining.**

***Index Terms*—Fully convolutional networks (FCNs), representation learning, self-supervised learning, semantic segmentation.**

## I. Introduction

In recent years, deep convolutional neural networks (CNNs) have been advancing the frontiers of many computer vision tasks such as image classification [1], [2] and semantic segmentation [3]–[8]. However, the performance of deep CNNs relies heavily on large amounts of labeled data. Data labeling requires intensive manual effort and is not even feasible for some applications. As a result, learning deep representation from unlabeled data has recently attracted great attention [9]. A promising strategy in this line of research is self-supervised learning which utilizes automatically generated labels for supervision. For example, Gidaris *et al.* [10] learn image representation by training an image rotation aware network. Such tasks (e.g., rotation prediction) in self-supervised learning are usually not the target task of interest; however, the representations learned during the process are still very useful. Such tasks are generally referred to as proxy or surrogate tasks in the current literature.

Many proxy tasks for representation learning have been proposed over the last few years. One such popular task is to exploit the spatial context within the visual images as the supervisory signal. For example, Doersch *et al.* [11] cropped a pair of neighboring patches and trained a network to predict their relative locations from the eight possible options. However, since the sampled patches have small sizes (e.g., $96 \times 96$), it is easy for one of the two patches to cover an area that contains little or no useful information, especially in high-resolution images. Using more patches could be a better choice for this problem. For example, the jigsaw puzzle system [12] used nine $80 \times 80$ patches. However, to extract features from multiple patches, that method adopts the Siamese network architecture where the training time increases significantly with the increase in the number of patches used.

Another drawback of current patch-based methods is that most of them use image classification as the main target task. Hence, they usually perform self-supervised learning on large-scale image classification datasets, such as the ImageNet [13], and then transfer the learned weights to other tasks, e.g., object detection. However, such an approach is suboptimal due to two reasons. First, training on a large-scale image classification dataset is time consuming, especially when graphic processing unit (GPU) memories are limited. In such cases, the batch size for training is usually limited to a small number, which would cost significantly more time than training with a large batch size. For example, it takes four weeks for Doersch *et al.* [11] to train their models on ImageNet. Second, features learned from image classification/dataset may not be suited for other tasks due to the difference in the data distributions. Given that

Manuscript received May 27, 2021; revised December 25, 2021 and April 2, 2022; accepted April 28, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62103137, Grant U2013203, Grant 61973106, and Grant U1913202; in part by the Project of Science Fund for Distinguished Young Scholars of Hunan Province under Grant 2021JJ10024; in part by the Key Research and Development Project of Science and Technology Plan of Hunan Province under Grant 2022GK2014; in part by the China Postdoctoral Science Foundation under Grant 2021M701156; and in part by the Project of Talent Innovation and Sharing Alliance of Quanzhou City under Grant 2021C062L. The work of Ajmal Mian was supported by the Australian Research Council Future Fellowship Award funded by the Australian Government under Project FT210100268. *(Zhengeng Yang and Hongshan Yu contributed equally to this work.) (Corresponding author: Hongshan Yu.)*

Zhengeng Yang, Hongshan Yu, Yong He, and Wei Sun are with the National Engineering Laboratory for Robot Visual Perception and Control Technology, College of Electrical and Information Engineering, Hunan University, Changsha 410082, China (e-mail: yuhongshancn@hotmail.com).

Zhi-Hong Mao is with the Department of Electrical and Computer Engineering and the Department of Bioengineering, University of Pittsburgh, Pittsburgh, PA 15260 USA.

Ajmal Mian is with the Department of Computer Science, The University of Western Australia, Perth, WA 6009, Australia.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3172423.

Digital Object Identifier 10.1109/TNNLS.2022.3172423

we can easily access massive amounts of unlabeled data for most applications, we believe that it is better to perform self-supervised learning on the same scenes of the target task to avoid the domain gap problem.

To learn feature representations for semantic segmentation with self-supervised learning, we incorporate the idea of Jigsaw [12] and relative location prediction [11] into a unified framework in this article. Highlights and contributions of this article are as follows.

1) Inspired by the patch-wise discrimination (real/fake) network adopted in image-to-image translation [14], we note that fully convolutional networks (FCNs) [15] can be approximately viewed as patch-wise classification networks. Based on this, we propose a self-supervised learning framework by formulating the jigsaw puzzle problem as a patch-wise index classification problem and then solve it with the FCN architecture. Our FCN-based method requires significantly fewer parameters than the standard Jigsaw method. Instead of predicting the permutation of nine patches, our patch-wise index classification aims to directly predict location indices of all patches at once. This reduces the number of parameters in the last layer of Jigsaw by 4.5 times.
2) By transferring our self-supervised models to the semantic segmentation task, we achieved a 5.8% point improvement on the mIoU compared with the baseline initialized with random values on Cityscapes dataset with only 503 training images. We use only about 1/6 training images of the Cityscapes to imitate practical scenarios where fully annotated images are usually limited to a small number.
3) We show that performing self-supervised learning and semantic segmentation on the same dataset can save efforts while leading to competitive results compared with performing these two tasks on different datasets. In particular, with a significant reduction (about four days) on the pretraining time, the performance of our method on PASCAL VOC even increased by 0.5% mIoU by replacing the ImageNet with PASCAL for self-supervised learning.

The rest of this article is organized as follows. The next section presents related work in self-supervised learning methods. Section III describes the theoretical analysis and implementation of our method in detail. Comprehensive ablation studies and experimental analysis are conducted in Section IV. Finally, conclusions are presented in Section V.

## II. Related Work

Existing self-supervised learning methods can be roughly divided into proxy task-based methods and contrastive learning-based methods. The contrastive learning-based methods typically learn representation by maximizing feature similarity between differently augmented views [16]–[18] or pretext transformations [19] of the same sample via a contrastive loss such as InfoNCE [20]. It is noted that maximizing the mutual information between related representations [21], [22] is also popular for contrastive learning. For example, Hjelm *et al.* [21] proposed to learn representation by maximizing the mutual information between global and local representations. Tiezzi *et al.* [23] extended the mutual information maximization to video frames selected with human-like focus attention. However, Purushwalkam Shiva Prakash and Gupta [24] demonstrated that existing contrastive learning methods benefit significantly from access to clean object-centric images that are difficult to obtain for semantic segmentation task. Thus, we mainly focus on the proxy task-based methods that are most relevant to our work when organizing our survey of the existing literature. Based on the proxy tasks, current self-supervised learning methods can be roughly divided into two large categories namely, context prediction and image generation.

### *A. Context Prediction*

Methods in the context prediction category try to exploit the internal spatial context within the visual data for supervisory signal. For example, Doersch *et al.* [11] used the eight possible relative locations between a pair of neighbor patches as the label for patch classification. Noroozi and Favaro [12] extended this idea of relative location prediction to solve a jigsaw puzzle problem with nine patches. The nine patches are shuffled randomly and then a network is trained to recover their order. Since the possible permutations of nine numbers is $9! = 362\,880$, it is not feasible to cover all permutation classes for classification with a deep network. Thus, a set of predefined permutations (such as 100) is used for the random patch shuffling. Hence, the Jigsaw problem is transformed into a 100-class classification problem. Later, Mundhenk Nathan *et al.* [25] improved these patch-based methods by incorporating numerous tricks, such as harnessing the jitters applied to the patches.

Our work also exploits context prediction for designing the proxy task. The most relevant works to our method are the relative location prediction [11] and the Jigsaw method [12]. As shown in Fig. 1, the relative location prediction method considers spatial contexts between only two patches of small sizes at a time. This has the potential problem that one of the patches may contain little or no useful information. In contrast, our method exploits contexts from nine or even 25 patches. The Jigsaw method also considers nine patches. However, it extracts features separately from each patch and then concatenates the features of nine patches for permutation classification. Hence, the number of parameters in the final layer are enormous. Instead of predicting the permutation of nine patches, our method directly predicts location indices of all patches at once.

### *B. Image Generation*

The image generation-based methods can be divided into two stages. In the first stage, part of an image is removed. The second stage then tries to recover or generate the removed part. For example, Pathak *et al.* [26] manually removed a region from an image and then trained a network to perform inpainting to recover the removed region using information from the remaining pixels. Another example of image generation is the commonly used automatic colorization task, i.e.,

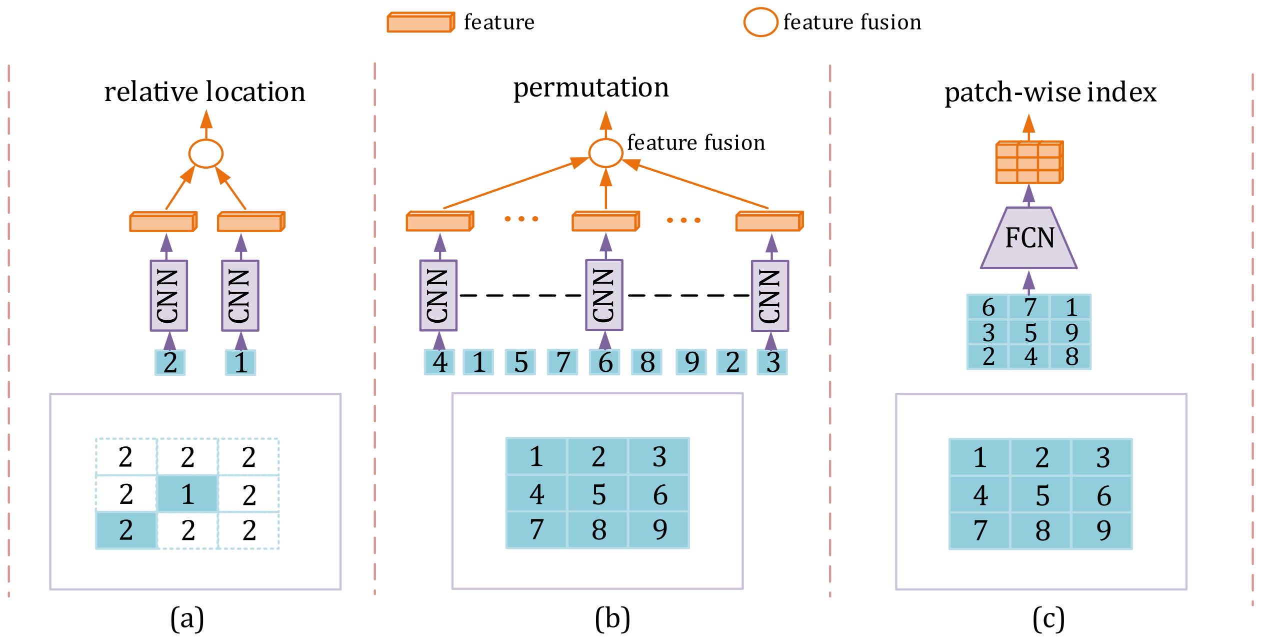


Fig. 1. Difference between our method and related methods. Current patch-based methods adopt the Siamese network to extract features from each patch independently. This leads to a bloated classification layer. Our method adopts an FCN to extract features from all patches simultaneously. This allows us to extend the number of patches to 25 without increasing the network complexity for better transfer learning. (a) Relative location prediction. (b) Jigsaw. (c) Ours.

recovering the three color channels from the remaining channel(s). Zhang *et al.* [27] and Larsson *et al.* [28] were among the first to use image colorization as a proxy task for representation learning. In particular, Zhang *et al.* [27] generated color channels from luminance, which was achieved by quantifying the Lab color space into a number of discrete intervals and then formulating the image colorization as a classification problem. Later the same authors [29] extended their idea to perform cross-channel generation, i.e., the remove/generate occurs between each pair of channels. Concurrently, Larsson *et al.* [30] replaced the commonly used AlexNet [31] with VGG [32] and introduced the hypercolumns [33] to improve the learning capacity of the colorization network. Compared with patch-based methods, colorization-based methods have an advantage that they do not change the spatial structure of the input. However, colorization-based methods may not learn color-based features which are important for many tasks such as semantic segmentation.

### *C. Other Methods*

Besides the two categories mentioned above, there are many other proxy tasks designed for self-supervised learning. For example, Dosovitskiy *et al.* [34] defined a series of exemplar classes for representation learning, where samples of each class were generated by applying different transformations to an image patch containing an obvious object. Gidaris *et al.* [10] defined four angler options (0, 90, 180, 270) for image rotation prediction. Based on this work, Feng *et al.* [9] added a nonparametric instance discrimination task to learn rotation irrelevant features. However, for the semantic segmentation dataset, it is difficult to meet the prerequisite that the sampled image patch contains an obvious object since there are usually many objects of different classes in an image. More recently, Jenni and Favaro [35] suggested that discriminative features can be learned by distinguishing real images from images with synthetic artifacts.

Exploiting relations between images for supervision is also popular. The deep-clustering [36] proposed to cluster deep features and then used the cluster assignments as pseudo labels for supervision. Similarly, Sabokrou *et al.* [37] exploited self-supervision derived from the fact that neighboring samples should share similar features.

Whereas the above mentioned methods are designed for static images, there are also many methods that leverage videos for self-supervised learning. Typical proxy tasks that exploit video data include temporal order prediction [38]–[40] and future frame prediction [41]. For example, Lee *et al.* [38] proposed to perform representation learning by sorting shuffled image sequences into the correct order. Then, Xu *et al.* extended the order prediction from image frames to video clips, to enable the learned representation for application to image analysis as well as video-based tasks such as action recognition [42]. The core idea of future frame prediction is to learn features by generating a video frame from its previous frames. However, video-based methods usually need to extract features from multiple frames during training, which could lead to huge GPU memory consumption.

## III. Method

In this section, we will first briefly review the core idea and architecture of FCN and then show that FCN can be approximately viewed as a patch-wise classification framework that can be used to solve the jigsaw puzzle problem.

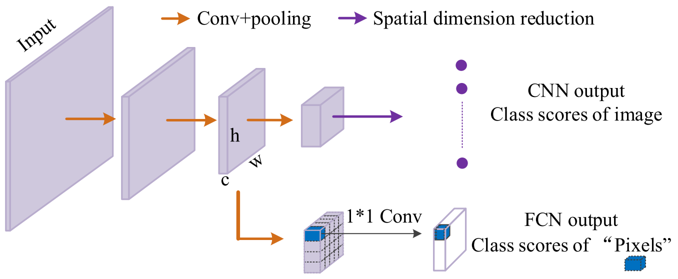


Fig. 2. Structures of CNN and FCN. CNN can be easily re-purposed to FCN by discarding the spatial dimension reduction layer and then performing classification at each spatial location.

## A. Fully Convolutional Network (FCN)

Given an image, typical CNNs convert it to a single feature vector and then predict its class. This is achieved by multiple "convolution + pooling" blocks and a separate spatial dimension reduction operation that usually presents as a fully connected layer (e.g., VGG [32]) or a global pooling layer (e.g., DenseNet [43]). To produce dense predictions, the FCN [15] discards the spatial dimension reduction operation and then makes prediction at each spatial location by applying a softmax function as

$$p_{c|x_{(i,j)}} = \frac{e^{w_c^T x_{(i,j)}}}{\sum_{b\in C} e^{w_b^T x_{(i,j)}}} \tag{1}$$

where $x_{(i,j)}$ is the feature vector at spatial location $(i, j)$, $C$ is the class set considered for the semantic segmentation task, and $w_c$ are the classification parameters of class $c$. The production of $w_c$ and $x_{i,j}$ is also called class score for class $c$ at spatial location $(i, j)$.

## B. Patch-Wise Classification With FCN

It can be easily seen in Fig. 2 that (1) is essentially a "pixel"-wise classification problem if each spatial unit of the output feature map of FCN is treated as a special "pixel." For convenience, we call these special "pixels" feature pixels. Next, we show that the FCN can be approximately viewed as a patch-wise classification framework by proving that the overlapping areas between input regions (or patches), where these feature pixels generated from, can be ignored to some degree.

Consider a FCN with $L$ layers and denote the feature map of layer $l$ as $f_l$, feature pixel at spatial coordinate $(i, j)$ of $f_l$ can be denoted by $f_l(i, j)$. Theoretically, the size of the input region where the feature pixel $f_l(i, j)$ generated from is equal to the receptive field (RF) size of $f_l$. In general, the RF size of a FCN can be computed by [44]

$$r = \sum_{l=1}^{L}\left((k_l - 1)\prod_{i=1}^{l-1} s_i\right) + 1 \tag{2}$$

where $k_l$ is the kernel size of layer $l$, $s_i$ is the output stride with respect to its previous layer. To obtain the exact input region corresponding to $f_l(i, j)$, we need to further compute the coordinate of the RF center. To this end, we define two auxiliary variables following [44]. The first one is effective

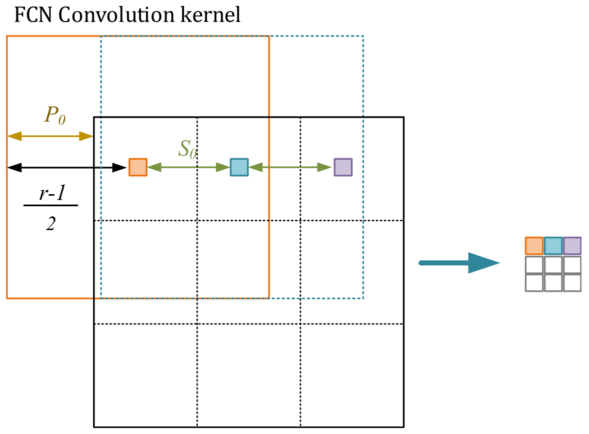


Fig. 3. Receptive field of FCN. FCN can be approximated as a convolution layer with kernel size equal to its receptive field $r$, stride equal to $S_0$, and padding number equal to $P_0$.

TABLE I
RECEPTIVE FIELD (RF) AND EFFECTIVE STRIDE OF COMMONLY USED FCNS. $S_0$ IS THE EFFECTIVE STRIDE WHEN TREATING FCN AS A CONVOLUTION LAYER

| | AlexNet | VGG16 | ResNet101 |
|---|---|---|---|
| RF | 195 | 212 | 1027 |
| $S_0$ | 32 | 32 | 32 |

stride: $S_0 = \prod_{l=1}^{L} s_l$, the stride of output feature map $f_L$ with respect to the input image. The second one is effective padding: $P_0 = \sum_{l=1}^{L} p_l \prod_{i=1}^{l-1} s_i$, the padding added to input image from the perspective of $f_L$, where $p_l$ is the padding added to $l$. With these two auxiliary variables, the coordinate of RF center of $f_L(i, j)$ can be computed by

$$(h_i, v_j) = \left(-P_0 + \frac{r-1}{2} + i\cdot S_0, -P_0 + \frac{r-1}{2} + j\cdot S_0\right). \tag{3}$$

In other words, we can view the entire FCN as a special convolution layer whose kernel size, stride and padding are equal to $r$, $S_0$, and $P_0$, respectively (see Fig. 3). While most basic CNNs (e.g., AlexNet, VGG16, ResNet) have five layers with stride equal to 2, $S_0$ of FCN built from these networks is typically equal to 32. On the other hand, according to (2), RFs of FCNs built from these commonly used networks are usually significantly larger than $2S_0$ (see Table I), which means that RFs of neighbor positions of FCN's output are highly overlapped. Nevertheless, Luo *et al.* [45] have shown that the distribution of impact within the RF is asymptotically Gaussian and the effective RF only takes up a small fraction of the full theoretical RF. Thus, it is reasonable to hypothesize that the effective region where each feature pixel generated from is a small patch around the center of the RF. Then, classification based on the output features of FCN can be approximately viewed as a patch-wise classification process.

## C. Self-Supervised Learning With FCN

Unlike predicting the permutations of patches (as in [12]), in real life, the jigsaw puzzle problem is typically solved by

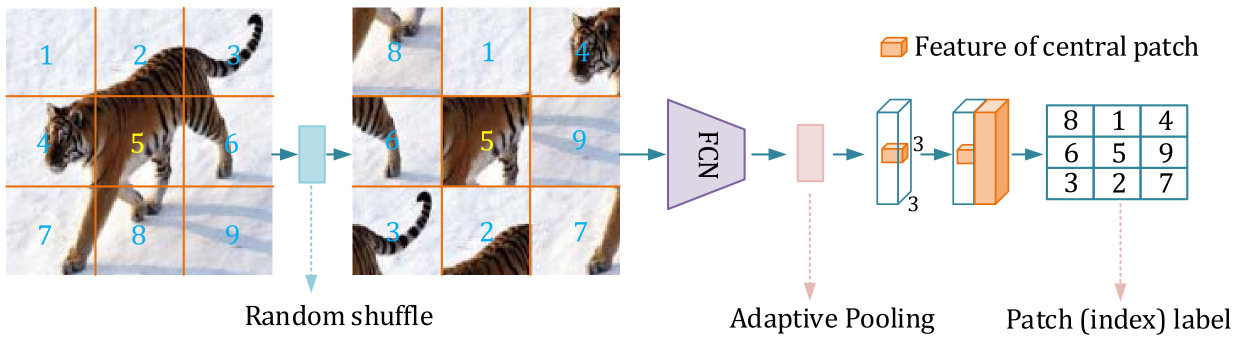


Fig. 4. Overview of our self-supervised learning framework. The basic idea of our self-supervised learning is to learn to predict the absolute location of patches in a jigsaw puzzle problem. Toward this goal, we first divide the training image into nine patches, which are then rearranged to a new image after a random shuffle operation. Then, a FCN, which we treat approximately as a patch-wise classification framework, is used to extract features for each patch. Finally, we concatenated features of each patch with features of the central patch, which act as the reference patch, for absolute location prediction.

finding the absolute location of each patch. In particular, when confronted with a jigsaw puzzle game, a human player tries to classify each patch into the correct location according to visual information extracted from all the patches. Thus, the jigsaw puzzle can be viewed as a patch-wise classification problem and this is exactly the approach we take in this article. More formally, denoting the number of patches of a jigsaw puzzle problem as $N$, and the feature vector of a patch $I_n$ as $x_n$, the jigsaw puzzle problem depicted in [12] can be written by

$$p(c|I_1, I_2, \ldots, I_N) = \text{MLP}\Big(\big[x_1, x_2, \ldots, x_N\big]\Big) \tag{4}$$

where MLP represents a function defined by a multilayer perceptron, $[\ldots]$ refers to the concatenation of feature vectors of different patches, $n$ represents the index of $n$th patch after a shuffle operation is performed over the $N$ patches, and $c$ is a label corresponding to a specific image order of a predefined permutation set. It can be concluded that (4) is essentially a sequence classification problem. While the jigsaw puzzle problem in practice is solved as

$$p(c|I_n, I_{s_1}, \ldots, I_{s_t}) = \text{MLP}\big(x_n, x_{s_1}, x_{s_2}, \ldots, x_{s_t}\big) \tag{5}$$

where $s$ represents indices of a number of patches that support human's decision on the location classification for the $n$th patch. For location classification of different patches, the number of support patches $t$ ($t \leq N-1$) may also be different. In this case, the $c$ is a label corresponding to one of the $N$ absolute positions of these patches. Compared with (4), there is no feature concatenation in (5) since the feature fusion of different patches conducted by human is still unclear.

Based on the above analysis on FCN and jigsaw puzzle problem, we propose to learn representations by solving a jigsaw puzzle problem with FCN. The overview of our method is shown in Fig. 4. First, an image is divided into a $3 \times 3$ grid which results in nine non-overlapping patches. These nine patches are then rearranged according to an order generated from a random shuffle operation. Finally, the rearranged image is fed into a FCN to predict the original position for each patch. To reduce the difficulty of the jigsaw puzzle problem, inspired by the relative location prediction in [11], we fix the central patch and use it as the reference for absolute location prediction of other patches. Thus, our FCN-based jigsaw puzzle can be formally written as

$$p\big(c|I_n, I_{\lceil \frac{N}{2} \rceil}\big) = \text{MLP}\big(\big[x_n, x_{\lceil \frac{N}{2} \rceil}\big]\big) \tag{6}$$

where $\lceil N/2 \rceil$ represents the central patch. Hence, our method can be also viewed as a relative location prediction problem. However, different from [11], which outputs only one relative location, our method outputs eight relative locations at once.

It can be observed from (4) and (6) that our network for Jigsaw problem is much more compact than the one used in the standard Jigsaw method. In particular, for the standard Jigsaw method, if the feature channel per patch is set to 512, the input of final convolution layer will contain $512 \times 9 = 4608$ channels. Thus, it will require $1 \times 1 \times 4608 \times 512 \approx 2.36$M parameters if the output feature dimension is to be reduced to 512 for final classification. In contrast, our method requires about 0.52M parameters using the above settings. On the other hand, a more compact network usually means fewer time cost for parameter gradient computation and fewer memory requirement for gradient storage.

## D. Network and Loss

Many state of the art CNN architectures exist (e.g., ResNet [1], DenseNet [43]) for representation learning. However, most CNNs require large batch size for training to ensure good performance of the batch normalization [46]. This leads to a huge consumption on GPU memory (especially for complex models like ResNet). Limited by the GPU resources at hand, we adopt a well-known lightweight model named MobileNetV2 [2] as the backbone for feature extraction. MobileNetV2 has many versions of different widths controlled by a parameter called "width multiplier." To use a large batch size for training, we set the width multiplier to 0.75 and further reduce the width of the last layer from 1280 to 512.

Similar to most FCN-based methods, we define the training loss using the cross entropy between prediction and ground truth. More formally, the training loss of our nine patch jigsaw

TABLE II

ABLATION STUDY OF OUR SELF-SUPERVISED LEARNING FOR SEMANTIC SEGMENTATION ON THE CITYSCAPES DATASET. THE BLOCKS REPRESENT THE FIVE CONVOLUTION LAYERS OF CNN AND THE VALUES ARE MIOU. FOR EXAMPLE, "BLOCK45" MEANS PARAMETERS OF BLOCK4 AND BLOCK5 ARE LEARNED FROM RANDOM VALUES (RANDOM COLUMN) OR SELF-SUPERVISED FEATURES (FINETUNE COLUMN), WHILE PARAMETERS OF OTHER BLOCKS (I.E., 1, 2, 3) ARE FROZEN WITH SELF-SUPERVISED FEATURES

| Self-supervised learning | | Semantic Segmentation | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Steps | ACC | block12345 | | block2345 | | block345 | | block45 | | block5 | |
| | | random | finetune | random | finetune | random | finetune | random | finetune | random | finetune |
| 20K | 60.6 | 42.0 | 43.4 | 42.1 | **44.7** | 41.9 | 44.0 | 41.0 | 41.9 | 26.8 | 27.5 |
| 30K | 85.1 | 42.0 | 43.2 | 42.0 | 44.3 | 42.2 | **44.8** | 42.2 | 43.2 | 26.8 | 27.2 |
| 50K | 92.8 | 42.0 | 42.8 | 42.6 | **44.6** | 42.1 | 43.4 | 42.4 | 43.0 | 26.5 | 27.0 |

puzzle problem is defined as

$$L = \frac{1}{9}\sum_{i=1}^{9} \underbrace{-\log \frac{e^{w_c^T x_{(i,j)}}}{\sum_{b=1}^{9} e^{w_b^T x_{(i,j)}}}}_{\text{cross\_entropy}} \tag{7}$$

where $c$ is the true location of patch $(i, j)$ and $x_{(i,j)}$ denotes the final feature of patch at location $(i, j)$.

### E. Datasets

In practical settings, we generally have only a small number of fully annotated samples for network training. On the other hand, massive unlabeled images are easily accessible from the Internet. Thus, we propose to use self-supervised learning to learn feature representations from the unlabeled data and then apply the learned representations for the downstream task in the same scene.

*Cityscapes:* Cityscapes [47] is an urban scenes dataset sampled from 50 European cities. It has 2975, 500, and 1525 fully annotated images for training, validation, and testing, respectively. To imitate real scenarios where fully annotated images are usually limited to a small number, we choose only 503 images (taken from Tubingen, Ulm, Weimar and Zürich) from the 2975 training images to train the segmentation network, and use the other 2472 images for self-supervised learning.

## IV. EXPERIMENTS

### A. Self-Supervised Learning as Target Task

We first treat our self-supervised learning as the target task to prove that the jigsaw puzzle problem can be solved by a FCN. Each sample of the training batch is a $576 \times 576$ image patch randomly sampled from an image augmented from the original Cityscapes dataset. The augmentations include random mirroring and random scaling that are commonly used in deep learning. We set the batch size to 36 and the training iterations to 50K. The learning rate is set to 0.1 and divided by 5 at 10K iteration and then decayed by a factor of ten every 10K iterations. We use the stochastic gradient descent (SGD) with momentum of 0.9 for parameter optimization.

Since our self-supervised learning is designed to solve a patch-wise classification task, we use the classification accuracy to evaluate its performance. Evaluation is performed over the 500 validation images of Cityscapes dataset. It is noted that we sampled only one patch for each image during this process, which means that we performed the Jigsaw task only 500 times during testing.

Training and evaluation were both implemented in Pytorch [48] and conducted on a computer equipped with a Titan X GPU, a CPU of Intel i7-6850k (six cores, 3.6 GHz), and 32 GB RAM. The experimental results are shown in Table II. We achieved an accuracy of up to 92.8% on the Jigsaw task, which supports that FCN is essentially a patch-wise classification network and can be used to solve the Jigsaw problem.

### B. Self-Supervised Learning as Proxy Task

We also validate the effectiveness of our self-supervised learning by performing a transfer learning task, i.e., use the learned features as pretrained weights for semantic segmentation.

Many works have proved that the self-supervised features at deeper layers are specific to the proxy tasks, and are hence not well suited for downstream tasks. To find the best transfer strategy between our self-supervised learning and semantic segmentation, we propose to freeze a subset of layers during the finetune process. In addition, the unfrozen layers were initialized either by random values or self-supervised features. Specifically, since most CNNs can be divided into five blocks according to the resolution of feature maps, we performed weights freezing using the "block" as unit (see Table II). It is noted that the final classification layer of segmentation was always initialized with random values since its dimension was different from the one used in self-supervised learning. To validate how the performance of proxy task influences the performance of segmentation task, we further performed transfer learning using parameters saved at different training steps. For convenience, we call these self-supervised models by PARAM-AT-N, where "$N$" is the training steps (iterations).

To train the segmentation network, we set the batch size to 8 and the training image size to $1024 \times 1024$. The total training steps was set to 30K. The learning rate was set to 0.05 and decayed by a factor of 10 at 5, 15, and 25K steps. Other training policies were similar to those used in the self-supervised learning.

We used the commonly used mean Intersection over Union (mIoU) metric to evaluate the performance of semantic segmentation. Results of our ablation experiments are shown in Table II. Note that we have used only about 1/6 training images of the Cityscapes in this article and did not use ImageNet pretrained weights for initialization in this experiment. Thus, the semantic segmentation performance is significantly lower than the one reported in the MobileNetV2 paper. From the Table II, important observations and conclusions can be drawn as follows.

1) The baseline method, where the entire feature extraction network was learned from random values with full supervision, achieved 42.0% mIoU on the Cityscapes validation set. When only block4 and block5 were learned from random values, our method, irrespective of the self-supervised models used, achieved comparable or better performance than the baseline. In addition, segmentation models initialized from our self-supervised features (finetune columns) always performed better than those initialized from random values (random columns). These results show that our self-supervised learning can learn useful feature representations.
2) Using PARAM-AT-30K for transfer learning, we obtained a 1.2% point improvement over the baseline when all feature blocks (block12345) were finetuned. The improvements achieved over the baseline increased first and then decreased with the increase in the number of frozen blocks. In particular, we arrived at the inflection point and obtained the largest improvement (2.8% point) over the baseline when block1 and block2 were frozen. However, the segmentation performance decreased significantly to 27.2% mIoU when block1 to block4 where all frozen (only block5 was not frozen). These results indicate that the low-level features learned by self-supervised learning are generic to downsteam tasks while high-level features are specific to the proxy task (Jigsaw in our case). It is noted that this and the following observations/conclusions are based on the results listed in the "finetune" columns of the Table II.
3) Compared to using PARAM-AT-30K, the inflection point of performance on target task came earlier when using PARAM-AT-50K. This suggests that the low-level features will also become specific to proxy task with the increase of training steps of self-supervised learning. The results obtained with PARAM-AT-20K seemed to conflict with this observation. In particular, the best performance achieved on target task with PARAM-AT-20K also came earlier when compared with results of PARAM-AT-30K. Nevertheless, since the reduction in mIoU were both larger than 1% point after the inflection points of PARAM-AT-30K and PARAM-AT-50K, we believe the inflection point of PARAM-AT-20K was the same, i.e., when only block345 were finetuned, with the one of PARAM-AT-30K.
4) While PARAM-AT-50K outperformed PARAM-AT-30K by 7.7% on the jigsaw puzzle task, it performed worse than PARAM-AT-30K on semantic segmentation in most cases. Thus, it can be concluded that the better performance of proxy task does not necessarily mean better performance for target task.

Overall, the above experiments strongly suggest that our self-supervised learning can learn useful features, especially at the low-level, for semantic segmentation.

## *C. Jigsaw Puzzle With More Patches*

Following the work of [12], we adopted nine patches for our Jigsaw-based self-supervised learning in the above experiments. In addition, these nine patches were sampled from an image of spatial size $576 \times 576$, which means each patch was $192\times192$. In practice, the number of patches of a jigsaw puzzle can be increased to hundreds or even thousands and the patch size can also be varied. Thus, to test if our method can learn better representations by solving a jigsaw puzzle with more patches, we increase the patches to 25 in this experiment. We also experimented with different patch sizes to evaluate how this influences the performance of our self-supervised learning.

For the self-supervised learning, we first adopted the same training image size as the one used in nine patches, leading to 25 patches of size $115\times115$. Then, we adopted the same patch size as the one used in nine patches and the training image size thus became $960 \times 960$. Note that each convolution layer of MobileNetV2 is followed by batch normalization, which can be formally written by

$$\hat{x_i} = \frac{x_i - \mu_i}{\sigma}\gamma + \beta \tag{8}$$

where $i$ was the index of feature channel, $\gamma$ and $\beta$ were two learnable parameters of linear transform, $\mu$ and $\sigma$ were the mean and standard deviation of a feature channel computed by

$$\mu_i = \frac{1}{m}\sum_{k\in S_i} x_k, \quad \sigma_i = \sqrt{\frac{1}{m}\sum_{k\in S_i}(x_k - \mu_i)^2 + \epsilon} \tag{9}$$

where $S_i$ is the set of pixels in which the mean and std are computed, $m$ is the size of this set, and $\epsilon$ was a small constant used to avoid the emergence of zero in the denominator of equation (8). It has been proven by many researchers [49] that the performance of batch normalization relies heavily on the batch size. We found that the batch size can only affect the size of pixel set $m$ in (9). In other words, the performance of batch normalization can be easily affected by the total number of training pixels. Thus, when the training image size was $960 \times 960$, we adjusted the batch size to 13 to ensure the total training pixels stay approximately the same to those used in other settings. All other training policies were kept the same with those depicted in Section IV-A.

When transferred to semantic segmentation, all the training policies were kept the same as depicted in Section IV-B and the self-supervised models used for transferring were "PARAM-AT-30K." The experimental results are reported in Table III and Fig. 5.

*1) Analysis:* As expected, given the same (image size) jigsaw puzzle task, the patch-wise classification accuracy decreased from 85.1% to 48.2% when the patch number increased from 9 to 25 with our FCN-based method. Nevertheless, the performance of segmentation task improved significantly in this case. For example, the segmentation accuracy improved by 4.9% points when block 1, 2, 3, 4 were frozen and the block5 was initialized from random values. This suggests that our Jigsaw-based self-supervised learning can learn better high-level semantic features with 25 patches. One reasonable explanation is that it becomes more difficult to

TABLE III

ABLATION STUDY ON PATCH NUMBERS AND SIZES WITH THE UNFROZEN BLOCK PARAMETERS INITIALIZED RANDOMLY OR FINE TUNED FROM OUR SELF-SUPERVISED MODEL

| Patches | sizes | ACC | block345 | | block45 | | block5 | |
|---|---|---|---|---|---|---|---|---|
| | | | random | finetune | random | finetune | random | finetune |
| $3 \times 3$ | 192 | 85.1 | 42.2 | 44.8 | 42.2 | 43.2 | 26.8 | 27.5 |
| $5 \times 5$ | 115 | 48.2 | 43.0 | 47.8 | 43.5 | 46.2 | 31.7 | 32.8 |
| | 192 | 56.6 | - | 46.5 | - | 44.8 | - | 33.1 |

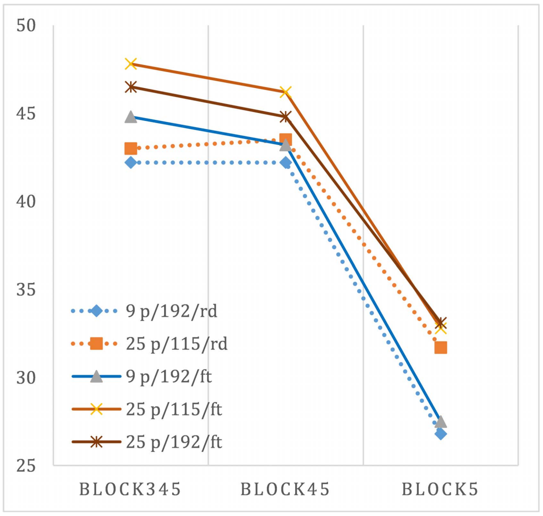


Fig. 5. Transfer learning accuracy (vertical axis) using our self-supervised models trained with different patch numbers and sizes. The dotted and solid lines mean that the parameters of unfrozen blocks (shown on the horizontal axis) are initialized from random values and self-supervised features, respectively.

exploit shortcut solutions to solve the jigsaw puzzle when more patches are used. The shortcut solutions may learn information highly specific to the jigsaw puzzle task but not the target task, i.e., the semantic segmentation. As mentioned by Noroozi and Favaro [12], the shortcuts to solve the jigsaw puzzle task mainly include low-level statistics, such as edge continuity, the pixel intensity/color distribution, and chromatic aberration. Among these, the Chromatic aberration, which is relatively difficult to understand, is the relative spatial shift between color channels that increases from the images center to the borders. Noroozi *et al.* proposed to avoid these three shortcuts using patch-independent normalization, overlapped patches, and color jittering, respectively. Instead of introducing these explicit strategies, our method can avoid these shortcuts implicitly. For example, note that the FCN was viewed approximately, but not exactly, as a patch-wise classification framework in our method. In other words, the patches of jigsaw puzzle in our method are implicitly overlapped although the overlapped area is limited. Nevertheless, the overlap area would increase with the increase of patch numbers given the same image size.

## D. Comparison to Other Methods

We finally evaluated our self-supervised learning on PASCAL VOC2012 to compare it with other methods. The original PASCAL VOC2012 training set contains only 1464 densely annotated images for semantic segmentation. Hariharan *et al.* [54] then expanded it to 10 582 images.

Following the common practice, we first performed the self-supervised learning on ImageNet [13], a well-known large-scale image classification dataset that contains about 1.3M images of 1000 natural classes, and then finetuned the learned features for semantic segmentation on PASCAL. Similar to most methods, we also adopt the AlexNet, which consists of five convolution layers and two fully connected layers, for feature learning in this experiment. However, it is better to perform the self-supervised learning and semantic segmentation on the same scene. To prove this hypothesis, we also conducted an experiment in which the self-supervised learning is performed on the 16 135 images included in the "test_JPEGImages" of PASCAL VOC2012.

*1) Implementation Details:* During self-supervised learning on ImageNet, we set the batch size to 128 and the training image size to $255 \times 255$. We train the model for 300K steps (about 30 epochs). The initial learning rate was set to 0.01 and then decayed by a factor of 10 at 100K and 200K steps. During self-supervised learning on PASCAL, we set the batch size to 64 and the training image size to $450 \times 450$. We train the model for 50K steps (about 200 epochs). The initial learning rate was set to 0.01 and then decayed by a factor of 10 at 20K and 40K steps. In the context of semantic segmentation, we trained the FCN32 model for 50K steps. The training batch was set to $36 \times 500 \times 500$. The initial learning rate was set to 0.01 and then decayed by a factor of 10 at 10K, 25K and 40K steps. Similar to [15], we also changed the padding number of the first convolution layer of AlexNet to 100 when transferring the learned features to semantic segmentation. The experimental results are reported in Table IV.

Since the methods listed in Table IV conducted training on different computational platforms, it maybe unfair to use the training time reported by these methods for comparison. On the other hand, it is cumbersome to reproduce all these methods to perform a fair comparison. Thus, we reproduced three methods, including the Jigsaw[1] (which is the most related to ours), RotNet[2] (backbone of the best-performing method), and Feng's method[3] (the best-performing method) using publicly available code and evaluate their training cost within our experimental environment. The pretraining time of these reproduced methods were estimated from the time cost on a 1-epoch training process. Note that Sabokrou *et al.* [37]

[1]https://github.com/bbrattoli/JigsawPuzzlePytorch
[2]https://github.com/gidariss/FeatureLearningRotNet
[3]https://github.com/philiptheother/FeatureDecoupling

TABLE IV

COMPARING OUR METHOD WITH SOTA METHODS ON PASCAL VOC2012 VALIDATION SET. * INDICATES THE USE OF DATA-DEPENDENT RE-SCALING METHOD PROPOSED BY KRÄHENBÜHL *et al.* [50]. THE MIOU VALUES WITH CITATIONS MEANS THEY ARE EXCERPTED FROM THE CORRESPONDING CITED PAPER. ALEXNET + BN REPRESENTS THAT EACH LINEAR LAYER OF ALEXNET IS FOLLOWED BY A BATCH NORMALIZATION LAYER

| Method | Backbone | Pre-training Setup | | | mIoU |
|---|---|---|---|---|---|
| | | Dataset | epoches | time cost | |
| ImageNet-Labels [31] | AlexNet | ImageNet(1.28M) | - | - | 48.0 |
| Random Gaussian | AlexNet | - | - | - | 19.8 [26] |
| Autoencoder | AlexNet | - | - | - | 25.2 [26] |
| Krähenbühl et al. [50] | AlexNet | ImageNet | - | - | 32.6 [51] |
| Inpainting [26] | ALexNet+BN | ImageNet | 10 | - | 30.0 |
| Counting [52] | AlexNet | ImageNet | - | - | 36.6 |
| Colorization [27]* | AlexNet+BN | ImageNet | - | - | 35.6 |
| Split-Brain [29]* | AlexNet+BN | ImageNet | - | - | 36.0 |
| Jigsaw [12] | AlexNet | ImageNet | 70 | 12d | 37.6 |
| Spot-Artifacts [35]* | AlexNet | ImageNet | - | - | 38.1 |
| Colorization [30] | AlexNet+BN+Hypercolumn | ImageNet(1.28M)+Places(2.4M) [53] | 10 | - | 38.4 |
| RotNet [10]* | AlexNet+BN | ImageNet | 30 | 8d | 39.1 |
| Mundhenk et al. [25] | AlexNet | ImageNet | 150 | - | 41.4 [9] |
| DeepCluster [36] | AlexNet | ImageNet | 500 | - | 45.1 |
| Feng et al. [9] | AlexNet | ImageNet | 200 | 50d | 45.3 |
| Ours (Random Gaussian) | AlexNet | - | - | - | 18.4 |
| Ours (ImageNet-Labels) | AlexNet | - | - | - | 50.0 |
| Ours (Feng et al.) | AlexNet | ImageNet | 4 | 1d | 37.2 |
| Ours (Jigsaw) | AlexNet | ImageNet | 5 | 1d | 32.5 |
| Ours | AlexNet | ImageNet | 30 | 5d | 37.5 |
| Ours | AlexNet | PASCAL VOC Test(16K) | 200 | 1d | 38.0 |
| Ours (Random Gaussian) | ResNet50 | - | - | - | 36.5 |
| Ours (ImageNet-Labels) | ResNet50 | - | - | - | 56.2 |
| Ours | ResNet50 | PASCAL VOC Test(16K) | 200 | 1d | 43.4 |

reported performance higher than the supervised method. However, we found a contradiction between the text description and segmentation performance shown in the Table II of their paper. In particular, Sabokrou *et al.* stated that “In all cases (except for the segmentation task) our results are superior to others by a considerable margin” when analyze the results shown in Table II. However, the Table II shows that the performance achieved over segmentation is more impressive than those achieved over other tasks. Thus, we did not add this article for comparison for the sake of prudence.

*2) Analysis:* When all the parameters were initialized from Random Gaussian values, which acts as the baseline in this experiment, the AlexNet-FCN32 achieved 18.4% mIoU on the PASCAL VOC2012 validation set under our training policy. In contrast, when all the parameters were pretrained with ImageNet classification labels, the AlexNet-FCN32 achieved the ceiling value 50% mIoU. Note that the ImageNet label pretrained model was downloaded from Internet. Our method achieved 37.5% mIoU on the PASCAL VOC2012 validation set with ImageNet-based pretraining, which outperformed significantly the baseline and the autoencoder method. Compared with state-of-the-art methods, our method outperformed Inpainting [26], Colorization [27], Split-Brain [29] by a large margin even without using batch normalization and data-dependent rescaling. Compared with the standard Jigsaw method, from which our method developed from, our method did not obtain an obvious improvement but rather a similar performance. Nevertheless, as stated in Section II, our FCN-based method is much easier to implement since it requires less memory to train. Feng *et al.* achieved the best performance (45.3% mIoU) among the compared methods. However, Feng *et al.* trained the feature extraction model on ImageNet for 200 epochs, a prohibitive training cost. In particular, it would cost about 50 days to perform self-supervised learning on ImageNet under our experimental environment. In contrast, with only one day cost on PASCAL pretraining, our method achieved 38.0% mIoU on PASCAL VOC2012 validation set. Similarly, with a significant reduction in the training cost, the performance of our method increased by 0.5% point after replacing the ImageNet with PASCAL for self-supervised learning, demonstrating that performing self-supervised learning and semantic segmentation on the same dataset can save efforts and at the same time lead to similar or even better results compared with performing these two tasks on different datasets. To further support this conclusion, we also trained the best-performing method (Feng’s method) and the method most related to ours (standard Jigsaw) on the ImageNet for one day. Compared with our PASCAL-based pretraining, Feng’s method and the standard Jigsaw lag 0.8% and 5.5% points behind our method, respectively.

To show that our method can be generalized to the widely used ResNet model, we applied our feature learning method to ResNet50 and then transferred it to semantic segmentation with FCN32 structure. Since the above experiments have verified that it is better to conduct self-supervised learning and semantic segmentation on the same dataset, we only performed self-supervised learning on the PASCAL VOC in this experiment. During the self-supervised learning stage, we change the batch size to 16 and the total training steps to 200K steps. For semantic segmentation, we change the batch size to 8 and the total training steps to 200K steps. As shown in the Table IV, our self-supervised learning method yielded a performance gain of 6.9% point with respect to the random initialized model. This experiment and the above experiments on AlexNet, MobileNetV2 strongly support that our self-supervised learning method can be applied to different datasets and models.

## V. Conclusion

In this article, we presented a novel self-supervised learning framework for semantic segmentation. We showed that the classical FCN can be approximately viewed as a patch-wise classification framework and applied to solve a jigsaw puzzle problem for representation learning. We achieved 5.8% mIoU improvement over the baseline model that was initialized from random values on Cityscapes validation set. Moreover, our method obtained competitive performance with the state-of-the-art methods on the PASCAL VOC2012 dataset with significantly smaller time costs on pretraining, which is achieved by directly performing self-supervised learning on the same scene on which semantic segmentation is to be performed.

## References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[2] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[3] Z. Yang *et al.*, "NDNet: Narrow while deep network for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 9, pp. 5508–5519, Sep. 2021.

[4] Z. Yang, H. Yu, M. Feng, W. Sun, and X. Lin, "Small object augmentation of urban scenes for real-time semantic segmentation," *IEEE Trans. Image Process.*, vol. 29, pp. 5175–5190, 2020.

[5] H. Yu *et al.*, "Methods and datasets on semantic segmentation: A review," *Neurocomputing*, vol. 304, pp. 82–103, Aug. 2018.

[6] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.

[7] H. Chen, Y. Jin, G. Jin, C. Zhu, and E. Chen, "Semisupervised semantic segmentation by improving prediction confidence," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 29, 2021, doi: 10.1109/TNNLS.2021.3066850.

[8] J. Jiang, J. Liu, J. Fu, X. Zhu, Z. Li, and H. Lu, "Global-guided selective context network for scene parsing," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 4, pp. 1752–1764, Apr. 2022.

[9] Z. Feng, C. Xu, and D. Tao, "Self-supervised representation learning by rotation feature decoupling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10364–10374.

[10] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," 2018, *arXiv:1803.07728*.

[11] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1422–1430.

[12] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 69–84.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1125–1134.

[15] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, Apr. 2017.

[16] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9729–9738.

[17] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[18] Y. Liu, Z. Li, S. Pan, C. Gong, C. Zhou, and G. Karypis, "Anomaly detection on attributed networks via contrastive self-supervised learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 5, 2021, doi: 10.1109/TNNLS.2021.3068344.

[19] I. Misra and L. van der Maaten, "Self-supervised learning of pretext-invariant representations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6707–6717.

[20] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[21] R. D. Hjelm *et al.*, "Learning deep representations by mutual information estimation and maximization," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[22] M. Tschannen, J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic, "On mutual information maximization for representation learning," in *Int. Conf. Learn. Represent.*, 2020.

[23] M. Tiezzi, S. Melacci, A. Betti, M. Maggini, and M. Gori, "Focus of attention improves information transfer in visual features," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 22194–22204.

[24] S. P. S. Prakash and A. Gupta, "Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 3407–3418.

[25] T. N. Mundhenk, D. Ho, and B. Y. Chen, "Improvements to context based self-supervised learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9339–9348.

[26] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2536–2544.

[27] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 649–666.

[28] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 577–593.

[29] R. Zhang, P. Isola, and A. A. Efros, "Split-brain autoencoders: Unsupervised learning by cross-channel prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1058–1067.

[30] G. Larsson, M. Maire, and G. Shakhnarovich, "Colorization as a proxy task for visual understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6874–6883.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[33] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 447–456.

[34] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1734–1747, Sep. 2016.

[35] S. Jenni and P. Favaro, "Self-supervised feature learning by learning to spot artifacts," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2733–2742.

[36] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 132–149.

[37] M. Sabokrou, M. Khalooei, and E. Adeli, "Self-supervised representation learning via neighborhood-relational encoding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8010–8019.

[38] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang, "Unsupervised representation learning by sorting sequences," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 667–676.

[39] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: Unsupervised learning using temporal order verification," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 527–544.

[40] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang, "Self-supervised spatiotemporal learning via video clip order prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10334–10343.

[41] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 843–852.

[42] Y. Zheng, Z. Liu, T. Lu, and L. Wang, "Dynamic sampling networks for efficient action recognition in videos," *IEEE Trans. Image Process.*, vol. 29, pp. 7970–7983, 2020.

[43] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[44] A. Araujo, W. Norris, and J. Sim, "Computing receptive fields of convolutional neural networks," *Distill*, vol. 4, no. 11, Nov. 2019.

[45] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," in *Proc. Adv. neural Inf. Process. Syst.*, 2016, pp. 4898–4906.

[46] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[47] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[48] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.

[49] Y. Wu and K. He, "Group normalization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.

[50] P. Krähenbühl, C. Doersch, J. Donahue, and T. Darrell, "Data-dependent initializations of convolutional neural networks," 2015, *arXiv:1511.06856*.

[51] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," 2016, *arXiv:1605.09782*.

[52] M. Noroozi, H. Pirsiavash, and P. Favaro, "Representation learning by learning to count," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5898–5906.

[53] R. Raturi, "Adapting deep features for scene recognition utilizing places database," in *Proc. 2nd Int. Conf. Inventive Commun. Comput. Technol. (ICICCT)*, Apr. 2018, pp. 487–495.

[54] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 991–998.

**Zhengeng Yang** received the B.S. and M.S. degrees from Central South University, Changsha, China, in 2009 and 2012, respectively, and the Ph.D. degree from Hunan University, Changsha, in 2020.

He was a Visiting Scholar with the University of Pittsburgh, Pittsburgh, PA, USA, from 2018 to 2020. He is currently a Post-Doctoral Researcher with Hunan University. His research interests include computer vision, image analysis, and machine learning.

**Hongshan Yu** received the B.S., M.S., and Ph.D. degrees in control science and technology from the College of Electrical and Information Engineering, Hunan University, Changsha, China, in 2001, 2004, and 2007, respectively.

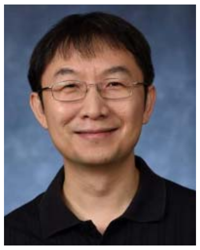
From 2011 to 2012, he worked as a Post-Doctoral Researcher with the Laboratory for Computational Neuroscience, University of Pittsburgh, Pittsburgh, PA, USA. In 2007, he joined Hunan University as an Assistant Professor and became a Professor in 2018. His research interests include autonomous robot and machine vision.

**Yong He** received the B.S. degree from the Anhui University of Science and Technology, Huainan, Anhui, China, in 2015, and the M.S. degree from the China University of Mining and Technology, Xuzhou, Jiangsu, China, in 2018. He is currently pursing the Ph.D. degree with Hunan University, Changsha, China.

He is also a Visiting Scholar with The University of Western Australia, Perth, WA, Australia. His research interests include computer vision, point clouds analysis, and deep learning.

**Wei Sun** received the M.S. and Ph.D. degrees in control science and technology from Hunan University, Changsha, China, in 1999 and 2002, respectively.

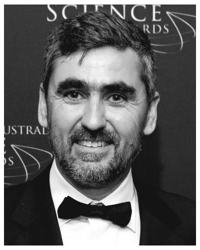
He is currently a Professor with Hunan University. His research interests include artificial intelligence, robot control, complex mechanical and electrical control systems, and automotive electronics.



**Zhi-Hong Mao** (Senior Member, IEEE) received the dual bachelor's degree in automatic control and applied mathematics and the M.Eng. degree in intelligent control and pattern recognition from Tsinghua University, Beijing, China, in 1995 and 1998, respectively, the S.M. degree in aeronautics and astronautics from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2000, and the Ph.D. degree in medical engineering and medical physics from the Harvard-MIT Division of Health Sciences and Technology, MIT, in 2006.

He joined the University of Pittsburgh, Pittsburgh, PA, USA, as an Assistant Professor in 2005 and became a Professor in 2018.

Dr. Mao was a recipient of the Outstanding Educator Award of the Swanson School of Engineering in 2009, the NSF CAREER Award in 2010, the Andrew P. Sage Best Transactions Paper Award of the IEEE Systems, Man and Cybernetics Society in 2010, the Outstanding Service Award as an Associate Editor of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS in 2013, and the Chancellor's Distinguished Teaching Award of the University of Pittsburgh in 2016.



**Ajmal Mian** (Senior Member, IEEE) is currently a Professor of computer science with The University of Western Australia, Perth, WA, Australia. His research interests include computer vision, machine learning, 3-D shape analysis, human action recognition, video description, and hyperspectral image analysis.

Prof. Mian has received three prestigious fellowships from the Australian Research Council (ARC) and several awards, including the West Australian Early Career Scientist of the Year Award, the Aspire Professional Development Award, the Vice-Chancellors Mid-Career Research Award, the Outstanding Young Investigator Award, the IAPR Best Scientific Paper Award, the EH Thompson Award, and the Excellence in Research Supervision Award. He has received several major research grants from the ARC, the National Health and Medical Research Council of Australia, and the U.S. Department of Defense. He serves as an Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON IMAGE PROCESSING, and *Pattern Recognition* journal.