TransFGU: A Top-Down Approach to Fine-Grained Unsupervised Semantic Segmentation

Zhaoyuan Yin[1,2], Pichao Wang[2(✉)], Fan Wang[2], Xianzhe Xu[2], Hanling Zhang[3(✉)], Hao Li[2], and Rong Jin[2]

[1] College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
zyyin@hnu.edu.cn
[2] Alibaba Group, Hangzhou, China
{pichao.wang,fan.w,xianzhe.xxz,lihao.lh,jinrong.jr}@alibaba-inc.com
[3] School of Design, Hunan University, Changsha, China
jt_hlzhang@hnu.edu.cn

Abstract. Unsupervised semantic segmentation aims to obtain high-level semantic representation on low-level visual features without manual annotations. Most existing methods are bottom-up approaches that try to group pixels into regions based on their visual cues or certain pre-defined rules. As a result, it is difficult for these bottom-up approaches to generate fine-grained semantic segmentation when coming to complicated scenes with multiple objects and some objects sharing similar visual appearance. In contrast, we propose the first top-down unsupervised semantic segmentation framework for fine-grained segmentation in extremely complicated scenarios. Specifically, we first obtain rich high-level structured semantic concept information from large-scale vision data in a self-supervised learning manner, and use such information as a prior to discover potential semantic categories presented in target datasets. Secondly, the discovered high-level semantic categories are mapped to low-level pixel features by calculating the class activate map (CAM) with respect to certain discovered semantic representation. Lastly, the obtained CAMs serve as pseudo labels to train the segmentation module and produce the final semantic segmentation. Experimental results on multiple semantic segmentation benchmarks show that our top-down unsupervised segmentation is robust to both object-centric and scene-centric datasets under different semantic granularity levels, and outperforms all the current state-of-the-art bottom-up methods. Our code is available at https://github.com/damo-cv/TransFGU.

Z. Yin—Work done during an internship at Alibaba Group.
P. Wang—Project lead.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-19818-2_5.

S. Avidan et al. (Eds.): ECCV 2022, LNCS 13689, pp. 73–89, 2022.
https://doi.org/10.1007/978-3-031-19818-2_5

1 Introduction

Given pictures of this world, can a semantic concept be deduced by certain prior rules, or can it be induced from the massive amount of observations? The answer to this question leads to different ways to obtain the semantic concept, and therefore brings different paradigms to the task of unsupervised semantic segmentation, which aims to obtain the pixel-wise classification as the semantic concept without any manual-annotated labels.

One way to tackle unsupervised image segmentation is to group low-level pixels into some semantic groups under the guidance of certain prior knowledge, *i.e.*the bottom-up manner [10,16,20,21,23,28,34], as shown in Fig. 1. Those methods often assume pixels in the same semantic object share a similar representation in the high-level semantic space. However, there is a large gap between low-level pixel and high-level semantic embeddings. Two semantically different objects may be mostly similar in low-level feature space, while the key to distinguishing them lies in some small areas reflecting the uniqueness of a semantic category. *e.g.*, the difference between a horse and a donkey may be found only on ears and legs. Accurate perception of these slight differences is the key to generating fine-grained segmentation, which is very hard to obtain by pixel-level deduction. In contrast, an object that has a large intra-class variance in appearance, *e.g.*the different parts of a person (head, arms, body...), may lead to dissimilar pixel-level features in different parts, which hinder the bottom-up methods from grouping these dissimilar features as an integrated object in high-level semantic space, due to the lack of high-level conceptual understanding of the whole object.

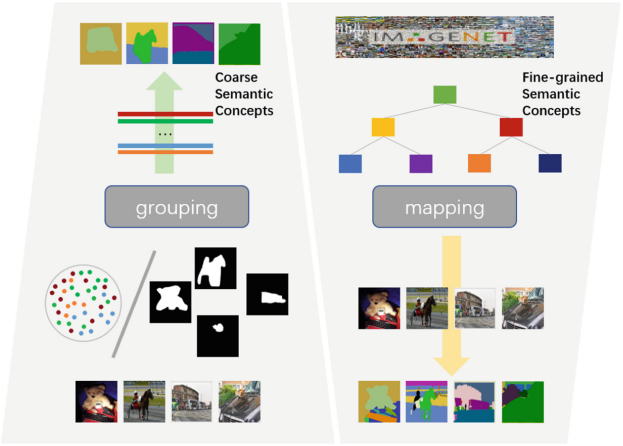

Fig. 1. Bottom-up (left) *vs.*Top-down (right) frameworks. Currently bottom-up manners group feature under the guidance of certain prior knowledge to form the semantic concepts (usually coarse), while our top-down manner maps the fine-grained structural semantic concepts obtained from ImageNet to pixel-level features and generates fine-grained segmentation.

To alleviate these problems, this work proposes a top-down approach to unsupervised semantic segmentation, as shown in Fig. 2. Instead of deducing high-level semantic concepts from low-level visual features, we start from the high-level fine-grained semantic knowledge induced from ImageNet. We benefit from the self-supervised learning method DINO [4] to gain the initial segmentation property of self-attention maps. The semantic representation obtained from DINO is more robust to the object appearance variations. Then we leverage the obtained prior to discover all the potential semantic categories presented in the target dataset, and group them into the desired number of semantic clusters according to their semantic similarity. It makes our method flexible to semantic concepts at different granularity levels. We then project the high-level semantic information to low-level pixel feature space by Grad-CAM [31,32,40], to generate fine-grained active maps for various semantic classes. The obtained active maps serve as pseudo labels for segmentation model training. A bootstrapping mechanism is employed to iteratively refine the quality of pseudo labels and gradually improve the final segmentation performance. The proposed method, named TransFGU, bridges the gap between high-level semantic representation induced by SSL and low-level pixel space, resulting in fine-grained segmentation in both object-centric and scene-centric images without any human annotations.

Through experiments on four public benchmarks, *i.e.*MS-COCO [26], PascalVOC [13], Cityscapes [11], and LIP [14], we show that our method can handle the cases of both foreground-only segmentation and foreground/background segmentation. Moreover, our method can control the semantic granularity level of segmentation by adjusting hyper-parameters, which overcome the limitations of previous methods that they can only produce coarse-level segmentation results. Our method achieves state-of-the-art results on all the benchmarks, surpassing all the current bottom-up methods.

In summary, our contributions are as follows: (i) We propose the first top-down framework for unsupervised semantic segmentation, which directly exploits the semantic information induced from ImageNet to tackle the fine-grained segmentation in an annotation-free way. (ii) We design an effective mechanism that successfully transfers the high-level semantic features obtained from SSL into low-level pixel-wise features to produce high-quality fine-grained segmentation results. (iii) Experiments show that our proposed method achieves state-of-the-art on a variety of semantic segmentation benchmarks including MS-COCO, Pascal-VOC, Cityscapes, and LIP.

2 Related Work

2.1 Self-supervised Representation Learning

Self-supervised representation learning has rapidly gained attention for its promising ability to represent the feature of semantic concepts without additional labels. It is also beneficial to many downstream tasks including detection, segmentation, and tracking. The common paradigm of self-supervised representation learning is to minimize the feature distance between two views of the same

object-centric image [3,4,7–9,15,17,24,39]. It allows learning semantic object concepts purely based on images from ImageNet without any annotations. The learned feature can perform well with KNN, which indicates the structural-semantic concepts have been exploited. BYOL [15] shows that representation learning without a single label can be on par or even better than its supervised counterpart. DINO [4], a recent work based on the Transformer, shows that the self-supervised learning method can learn a good representation of structured semantic category information, which has a nice property of focusing on the area of foreground objects. While most of them aim to learn the overall representation from the object-centric image, some others [25,30,35–38] tend to generate pixel-level dense features, which better benefit the tasks that require dense matching like segmentation or detection. However, all of these methods tend to learn the pixel-level correspondence on the category of objects that appear in different views of an image rather than learning the pixel-level semantic concept, so the learned representation embedding feature cannot convert to segmentation mask directly without additional manual labels.

2.2 Unsupervised Segmentation

Most of the current works proposed to tackle unsupervised segmentation start from the observation of pixel-level features and grouping the pixels into different semantic groups with various priors rules that are independent of the training data. [21,23] group the similar pixel extracted from a randomly initialized CNN in both embedding and spatial space while keeping the diversity of embedding features. [16,20,28,29] maximize the mutual information between the pixel-level feature of two views from the same input image to distill the information shared across the image. PiCIE [10] disentangles features between different semantic objects by leveraging two simple rules, *i.e.*invariance to geometric and equivariance to photometric while transforming two different views of an image and its feature by two asymmetric augmentation processes. These rules often fail to tackle the case with complex scenes, where objects of different categories might share similar appearances and objects of the same category might have a large intra-class appearance variation. Unlike the bottom-up manner, our method is based on the top-down pipeline that obtains fine-grained semantic concepts prior information before performing the segmentation, which is crucial to tackling these two problems.

Besides the bottom-up methods, several works use intermediate features as additional cues to guide the semantic feature grouping, including saliency map [1,2,6,34], super-pixel [21,28], and shape prior [19,22]. MaskContrast [34] leverages two additional unsupervised saliency models to generate pseudo labels of the foreground object in each image and train the final segmentation model with a two-step bootstrapping mechanism, and then cluster all the gathered foreground masks to form the semantic concept by contrast learning. However, these methods assume that objects in an image are salient enough, which is not always true in some complicated scene-based cases and could lead to low-quality segmentation.

3 Methods

Given a set of images $I = [I_1, ..., I_N] \in [0,1]^{N\times 3\times H\times W}$ with total N samples, our goal is to generate a group of K segmentation probability maps for each image, denoted as $P = [P_1, ..., P_N] \in [0,1]^{N\times K\times H\times W}$, where K indicates the predefined number of semantic categories to be segmented in I, and (H, W) are the height and width of the image, respectively. To achieve this goal, a semantic prior model self-supervised trained on ImageNet is introduced to extract the potential top-level semantic information from the whole set of images I, and the extracted top-level semantic features are clustering into K groups based on their semantic similarity, resulting in K semantic concept features. Then, for each $I_i \in I$, a top-down semantic information mapping pipeline is introduced to generate pixel-level semantic pseudo label according to K semantic concept features based on Grad-CAM. An encoder is needed to extract pixel-level features, and a decoder is required to convert the pixel-level features into P_i corresponding to the generated semantic pseudo label. In this section, we will first introduce some basics of the encoder and decoder, then describe the designed top-down pipeline, followed by a bootstrapping process that helps further refine the results.

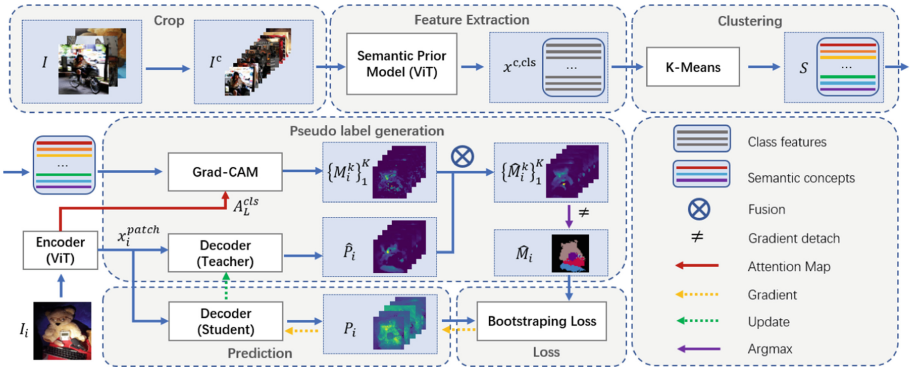

Fig. 2. Our proposed top-down pipeline for unsupervised semantic segmentation. The training samples are first cropped and resized into square patches using sliding windows, and class features for each cropped patch are extracted by ViT pretrained on DINO. The obtained class features are clustered into semantic concept representations, which are mapped to the pixel level to generate the pseudo labels. The quality of pseudo labels is further bootstrapped by the teacher network, based on which a student network is trained to refine the semantic knowledge. The teacher network is updated by the learned student network to produce better pseudo labels for the next round of training.

3.1 Revisiting ViT and Segmenter

Previous works on SSL have shown that ViT [12] models pretrained on ImageNet using methods like DINO [4] can provide surprisingly good properties for

segmentation tasks due to its explicit semantic information learned during SSL. Thus, the pre-trained ViT is employed as ENCODER$(\cdot)$ in our pipeline, for its capability of providing high-level semantic priors through the class token, as well as extracting pixel-level features in image patch tokens. Next, we will introduce some basic notations and terminologies about ViT for easier reference.

ViT is an attention-based model with L layer attention blocks. Each attention block at the l-th layer produces an attention map $A_l \in [0,1]^{(1+hw)\times(1+hw)}$ that can be formulated as:

$$A_l = \text{SOFTMAX}(\frac{Q_l \cdot K_l^{\mathrm{T}}}{\sqrt{d'}}) \tag{1}$$

where $Q_l, K_l \in R^{(1+hw)\times d'}$ are the query and key, respectively, which are mapped and reshaped from the layer input x_l, d' is the dimension of embedding feature in attention block, and (h, w) is the size of the feature map.

As an ENCODER$(\cdot)$, the final outputs of ViT are denoted as class token x^{cls} and image patch tokens as x^{patch}. It is worth noting that, ViT pretrained with DINO is more irreplaceable for generating high-level semantic priors through class token, however, the model for generating pixel-level features can be substituted by other networks capable of extracting nice local features. Here we unify the two networks as the same ViT model for simplicity.

For the decoder which generates segmentation probability maps using the output of encoder, there are also lots of options while transformer-based methods are preferable due to their robustness to noises. Segmenter [33] is selected among them for its simplicity and straightforward interface to take x^{patch} as input.

Segmenter is a transformer-based segmentation model which consists of multiple cross-attention layers. It takes x^{patch} as input, and the output probability mask is:

$$\text{MASK}(x', C) = \frac{x' C^{\mathrm{T}}}{\sqrt{d}} \tag{2}$$

where x' and $C \in \mathcal{R}^{K\times d}$ are the patch embedding and class embedding output by the final layer in the segmenter, respectively. $\text{MASK}(x', C) \in R^{hw\times K}$ is the downsampled probability maps for K categories, and then it is upsampled to (H, W) to obtain the final mask P. This whole process is denoted as DECODER$(\cdot)$ in the following sections.

3.2 Top-Down Pipeline

To generate semantic mask P, a ViT model obtained by DINO is first used as a semantic prior model to encode all the potential semantic information in I. In the real-world application, an image might contain abundant and complex semantic concepts instead of a simple object, therefore, applying ViT encoder to the whole image might not be sufficient to attend to all potential semantic priors. Instead, we propose to apply a sliding window cropping operation to each image. For every image I_i, n_c square patches are generated, each with

side length $\beta \times \min(H, W)$ where β is a scaling factor taking a series of values. Patches from all images are put together to form a larger "patch image" set $I^c = [I_1^c, \ldots, I_N^c]$ with I_i^c representing the patch set for image I_i. All patches are resized to $(\frac{\min(H,W)}{2}, \frac{\min(H,W)}{2})$ and treated as a whole image before feeding into the ViT encoder.

Class features and patch features are extracted from I^c, denoted as $x^{c,cls} = [x_1^{c,cls}, \ldots, x_N^{c,cls}]$ and $x^{c,patch} = [x_1^{c,patch}, \ldots, x_N^{c,patch}]$, in which:

$$(x_i^{c,cls}, x_i^{c,patch}) = \textsc{Encoder}(I_i^c) \tag{3}$$

here the ViT encoder is served as a semantic prior model, $x^{c,cls}$ can be regarded as containing all the potential semantic concepts that have appeared in I, $x^{c,patch}$ is discarded because of its weak semantic prior property. Assuming there are in total K pre-defined semantic categories, we can obtain a set of semantic feature $S = [S_1, \ldots, S_K] \in \mathcal{R}^{K \times d}$ by applying K-means on $x^{c,cls} \in \mathcal{R}^{(N \times n_c) \times d}$ over all the $(N \times n_c)$ features based on Euclidean distance. K can be set as the number of classes, to generate segmentation results at different desired granularity levels.

The extracted semantic features $S_k \in S$ can be seen as a category-wise pseudo semantic label corresponding to a certain granularity level of semantic. Next, the Grad-CAM [31] is adopted to visualize the corresponding category-specific response area to the target pseudo semantic label S. The response heat-map is treated as a coarse semantic area that locates the target semantic object in the pixel-level feature space. Furthermore, these coarse semantic areas are treated as pseudo labels $M = [M_1, \ldots, M_N]$, $M_i \in [0, K]^{H \times W}$, and a decoder will be trained based on these pseudo labels.

To be more specific, inspired by [5], we first obtain the class token x_i^{cls} and patch token x_i^{patch} of the whole image I_i by ViT encoder:

$$(x_i^{cls}, x_i^{patch}) = \textsc{Encoder}(I_i), \tag{4}$$

then take attention map from *cls* token of the last attention block, and take the feature map while discarding the *cls* token itself. Denote $A_L^{cls} \in [0, 1]^{h \times w}$ as the feature map, gradient is generated on A_L^{cls} w.r.t S_k by maximizing the cosine similarity between x_i^{cls} and each $S_k \in S$:

$$\min(1 - \frac{x_i^{cls} S_k^{\mathrm{T}}}{\sqrt{d}}). \tag{5}$$

We take the gradient value of all the patch locations on A_L^{cls}, and add it back to A_L^{cls} to generate K response maps $[M_i^1, \ldots, M_i^K] \in \mathcal{R}^{K \times h \times w}$, then the pseudo label M_i can be obtained by appling $\arg\max$ operator to each patch location (h', w') across all the K response maps, where $h' \in [0, h], w' \in [0, w]$, and upsample to the target size of (H, W):

$$M_i = \textsc{Upsample}(\arg\max_{(h',w')}(M_i^1, \ldots, M_i^K)), \tag{6}$$

$$M_i^k = \textsc{Grad-CAM}(x_i^{cls}, S_k | A_L^{cls}) + A_L^{cls}. \tag{7}$$

If only the foreground objects need to be segmented in an image, we calculate the background probability M_i^{bg} based on all the foreground pseudo labels as follows:

$$M_i^{bg} = \text{RELU}(T^{bg} - \max_{k \in [0,K]} M_i^k) \tag{8}$$

in which T^{bg} is a hyper-parameter that represents the max probability of background, and is set to 0.1 in our experiments. M_i^{bg} is then upsampled and concatenated with M_i. The decoder can be trained by standard cross-entropy loss with its output denoted as segmentation probability maps P:

$$\mathcal{L} = \text{CE}(P, M). \tag{9}$$

3.3 Bootstrapping

Training the decoder directly with the initial pseudo labels may lead to segmentation of low quality because the noise level in pseudo labels can be detrimental. For example, two categories with similar high-level semantic meaning may generate similar response maps on the same image, thus disturbing the ranking of K values on the same location of the pseudo labels and further misleading the learning process. Additionally, the boundary produced by the pseudo labels is not precise enough for certain semantic objects due to the coarse response area in response maps generated by Grad-CAM.

We tackle these problems by a bootstrapping mechanism. First, a teacher-student framework is introduced to refine the pseudo labels progressively. Second, we introduce a set of loss functions to force the model to learn discriminative abilities and prevent the model from over-fitting the noise in the pseudo labels.

Teacher-Student Network. In our design, both teacher and student network have exactly the same architectures as DECODER$(\cdot)$. As shown in Fig. 2, given an initial pseudo mask M generated by Grad-CAM, its bootstrapped version $\hat{M} = [\hat{M}_1, \ldots, \hat{M}_N]$ can be obtained by aggregating M and the output of the teacher network prediction $\hat{P} = [\hat{P}_1, ..., \hat{P}_N] \in [0,1]^{N \times K \times H \times W}$:

$$\hat{P}_i = \text{TEACHER}(x_i^{patch}), \tag{10}$$

$$\hat{M}_i = \text{UPSAMPLE}(\arg\max_{(h',w')}(\hat{M}_i^1, \ldots, \hat{M}_i^K)), \tag{11}$$

$$\hat{M}_i^k = 0.5 \cdot (M_i^k + \hat{P}_i^k), k \in [1, \ldots, K]. \tag{12}$$

where $\hat{P}_i^k \in \mathcal{R}^{h \times w}$ is the k-th probability map in $\hat{P}_i$ before upsampled. With this bootstrapped pseudo labels, the student network is trained with a cross-entropy loss $\text{CE}(P, \hat{M})$ with P as the output segmentation probability of student network, *i.e.*$P_i = \text{STUDENT}(x_i^{patch})$.

In the next round, the teacher network is updated with the parameter trained from the student network, so it can produce a better prediction of $\hat{P}$, which further improves the bootstrapped pseudo labels $\hat{M}$. The student network in the

next round is reset to the initial state and trained with the improved $\hat{M}$. Therefore, the quality of bootstrapped pseudo labels and the output of the student network can be improved progressively as the iteration goes on.

Bootstrapping Loss. A few additional loss functions are further proposed to provide better guidance for training the decoder based on the pseudo label masks.

First, we observe that even though categories with similar semantic meanings are difficult to differentiate thus might confuse the training process, categories with much different semantic meanings are actually easier to identify with high accuracy. Therefore, in addition to the cross-entropy loss which aligns probability masks P and the "correct" pseudo labels $\hat{M}$, we also introduce a set of "wrong" pseudo labels $\hat{M}'$ so that its cross-entropy loss with P should be maximized. Thus a peer loss [27] is defined as below:

$$\mathcal{L}_{peer} = \mathrm{CE}(P, \hat{M}) - \alpha \cdot \mathrm{CE}(P, \hat{M}'). \tag{13}$$

where α is a hyper-parameter, and $\hat{M}'$ is the constructed negative labels by shuffling the K pseudo labels of $\hat{M}$.

Second, it is observed that categories with similar semantic meanings usually generate similar response values at the same location of the pseudo label masks, making it hard to distinguish the correct category from the others and further slow down the training. An uncertainty loss is introduced to diminish such uncertainty. Specifically, this is achieved by maximizing the gap between the largest and the second largest value at each location of the output probability map P:

$$\mathcal{L}_{unc} = 1 - \frac{1}{hw} \sum_{(h',w')} (p' - p''). \tag{14}$$

in which p', p'' are the largest and second largest probability value among the K probabilities at location (h', w').

Third, it is beneficial to keep the representations diversified in the decoder model to learn more meaningful categories information. This is done by a diversity loss which maximizes the summation of all the pairwise distances between the class embeddings of K categories, *i.e.*, minimizing their cosine similarities:

$$\mathcal{L}_{div} = 1 + \frac{1}{K^2} \sum \frac{C \cdot C^T}{\sqrt{d}}. \tag{15}$$

where C is the class embedding in the decoder as in Eq. 2.

The final loss is:

$$\mathcal{L} = \mathcal{L}_{peer} + \omega_1 \cdot \mathcal{L}_{div} + \omega_2 \cdot \mathcal{L}_{unc}. \tag{16}$$

in which ω_1, ω_2 are hyper-parameters to balance between different losses, and set to 1 and 0.3 in our experiments, respectively.

4 Experiments

4.1 Dataset and Evaluation Metrics

We conduct experiments on four semantic segmentation benchmarks including COCO-Stuff, Pascal-VOC, Cityscapes, and LIP, with different definitions of semantic concepts.

COCO-Stuff. COCO-Stuff is a challenging benchmark with scene-centric images, which contains 80 things categories and 91 stuff categories. The total number of training and validation samples is 117,266 and 5,000 images, respectively. Note that the previous works [10,20] only conduct their experiments on the "curated" data (2175 images total) in which the stuff occupied at least 75% pixels of an image. We evaluate our method on three settings: 1) COCO-S-27:all categories in COCO-Stuff merged into 27 superclasses as [10], 2) COCO-S-171: the original 171 things and stuff categories, and 3) COCO-80: only 80 things categories without background stuff.

Cityscapes. Cityscapes contain 5,000 images focusing on street scenes, in which 2,975 and 500 images are used for training and validation. All pixels are categorized into 34 classes. We follow [10] to merge 34 classes into 27 classes after filtering out the 'void' class.

Pascal-VOC. Pascal-VOC contains 1464 images for training and 1,449 for evaluation, with 20 classes as foreground objects to be segmented and the rest pixels as background.

LIP. LIP contains person images cropped from MS-COCO [26]. It has 30,462 training images and 10,000 validation images, with 19 semantic parts defined. The 19 categories are merged into 16 and 5 coarse categories to evaluate the ability of our method to handle the semantic concepts of different granularities. Please refer to supplementary materials for more details.

Evaluation Metric. Following the standard evaluation protocols for unsupervised segmentation used in [10,20,34], we use Hungarian matching algorithm to align indices between our prediction and the ground-truth label over all the images in the validation set. Two metrics are reported for comparison, *i.e.*Intersection over Union (mIoU) and Pixel Accuracy over all the semantic categories.

4.2 Implementation Details

Cropping and Evaluation Protocol. The scaling factor β in sliding window cropping is set to $0.5, 0.4, 0.3, 0.2$, and the step size of the sliding window is set to $\frac{1}{2} \times \beta \times \min(H, W)$.

Foreground prior is introduced in the cropping operation, which is defined as the binarized attention map $\bar{A}_L^{cls}$ obtained from A_L^{cls} on the last attention block by setting the values greater than the mean value of A_L^{cls} as 1 and the rest as 0. Cropped patches are separated into foreground patches and background patches. A patch is a foreground patch when it has more than 50% of pixels belonging to $\bar{A}_L^{cls}$, and a background patch when it contains more than 80% of pixels that belong to $(1 - \bar{A}_L^{cls})$. K-Means is executed on these two groups of patches separately. If a patch is treated as a foreground patch, its probability in those background categories will have a default value of 0, and a background patch is treated in a similar way. The separation of foreground and background patches help generate more accurate semantic clusters and better pseudo labels.

Different cropping protocols are applied to the four datasets, due to their different properties and requirements. On Cityscapes, foreground prior is not involved as there is no definition of foreground/background. On LIP, the images are obtained as bounding boxes around each person without much background area, so we treat all the areas as foreground. On COCO-80, Pascal-VOC and LIP, Eq. (8) is used to generate background probability.

Trainning Setting. We use *ViT-small* 8×8 pre-trained on ImageNet by DINO [4] as encoder and fix the weight during training, and Segmenter [33] with random initial weights as the decoder. We pre-compute and save all the initial pseudo labels and build a pseudo label bank which considerably accelerates the training. More training details are included in the supplemental materials.

Data Augmentation. A set of data augmentations are utilized, such as color jittering, horizontal flipping, Gaussian blur, color dropping, and random resized crop following [4,15]. The cropped image is resized to $(\frac{H}{2}, \frac{W}{2})$ for the following training. We crop and resize the initial pre-computed pseudo label to $(\frac{h}{2}, \frac{w}{2})$ by RoI-Align operator [18].

4.3 Main Results

Baseline. We compare our method with PiCIE [10], IIC [20] and MaskContrast (MC) [34], which can generate segmentation masks without further fine-tuning. Since IIC and PiCIE cannot distinguish foreground objects from the background, they can only be applied on COCO-Stuff and Cityscapes. MaskContrast is only able to deal with the foreground object without background area, so it is only applied to COCO-80 and Pascal-VOC. None of these methods can actually work well on LIP which is essentially a task of fine-grained human parsing. Note that our method can be successfully adapted to all these datasets.

Table 1. Results on four benchmarks. * indicates the results are evaluated on the "curated" samples. † denotes PiCIE trained without auxiliary clustering.

Dataset	Method	mIoU	Acc.	Dataset	Method	mIoU	Acc.
COCO-S-27*	IIC [20]	6.71	21.79	COCO-80	MC [34]	3.73	8.81
	PiCIE† [10]	13.84	48.09		TransFGU	**12.69**	**64.31**
	PiCIE [10]	14.36	49.99	Cityscapes	IIC [20]	6.35	47.88
	TransFGU	**17.47**	**52.66**		PiCIE [10]	12.31	65.50
COCO-S-27	IIC [20]	2.36	21.02		TransFGU	**16.83**	**77.92**
	PiCIE [10]	11.88	37.20	Pascal-VOC	MC [34]	35.00	79.84
	TransFGU	**16.19**	**44.52**		TransFGU	**37.15**	**83.59**
COCO-S-171	IIC [20]	0.64	8.67	LIP-5	TransFGU	25.16	65.76
	PiCIE [10]	4.56	24.66	LIP-16	TransFGU	15.49	60.08
	TransFGU	**11.93**	**34.32**	LIP-19	TransFGU	12.24	42.52

Quantitative Evaluation. The comparison is shown in Table 1. Our method exceeds all the baseline methods on mIoU and pixel accuracy of all datasets. Our method has been adapted to different granularity levels in the same dataset, *e.g.*27/171 categories on COCO-S and 5/16/19 categories in LIP, to make a fair comparison with other methods which mainly work on coarser-level categories. Our method shows a larger margin of performance improvement for the settings with much finer-grained categories. Besides, it can also be adapted to only segment foreground objects, *i.e.*COCO-80, Pascal-VOC, and LIP, demonstrating its all-around flexibility.

Fair Comparison with Prior Arts. The difference in backbone architecture and the pre-training manner to obtain backbone weights influences the model performance. For fair comparison, we conduct the following experiments successively: (1) reproduce IIC with the ResNet-18 backbone fully-supervised trained on ImageNet to align the amount of training data; (2) reproduce IIC and PiCIE with ResNet-50 and ViT-S fully-supervised trained on ImageNet instead of its original ResNet-18 to make backbone parameters comparable with our method; (3) reproduce IIC, PiCIE and MaskContrast with the ResNet-50 backbone trained by DINO to unify the training manner; (4) reproduce the PiCIE and MaskContrast with the ViT backbone trained by DINO instead of ResNet-50 to eliminate the impact of the architecture difference. To obtain the input feature maps of FPN used in PiCIE, we extract pixel features every three attention blocks amount 12 layers and resize them to the target size. The results are shown in Table 2. Our method outperforms all the baseline in each case. Note that the performance of IIC and PiCIE with backbones trained by SSL manner becomes inferior to its original fully-supervised version. It might be due to the feature learned by SSL containing more fine-grained foreground details, which may distract the bottom-up method to find similar features that belong to the same category.

Table 2. Results for the effectiveness of pretrained weight. *, † and ‡ indicates the weights are trained on ImageNet by fully-supervised, DINO and MoCo, respectively.

Dataset	Method	mIoU	Acc.	Dataset	Method	mIoU	Acc.
COCO-S-171	IIC-R18 [20]	0.64	8.67	COCO-S-171	PiCIE-R50† [10]	2.30	13.50
	IIC-R18* [20]	1.22	13.92		PiCIE-ViT† [10]	3.02	18.45
	IIC-R50* [20]	2.15	15.72		TransFGU†	**11.93**	**34.32**
	IIC-R50† [20]	0.98	11.89	Pascal-VOC	MC-R50‡ [34]	35.00	79.84
	PiCIE-R18* [10]	4.56	24.66		MC-R50† [34]	28.72	78.72
	PiCIE-R50* [10]	5.61	29.79		MC-ViT† [34]	31.24	79.18
	PiCIE-ViT* [10]	6.82	31.17		TransFGU†	**37.15**	**83.59**

Qualitative Evaluation. Some qualitative comparisons on different benchmarks are shown in Fig. 3. Our results can show richer semantic information in both scene/object-centric images on all datasets. We obtain much detailed segmentation on foreground objects, especially in complicated scenarios.

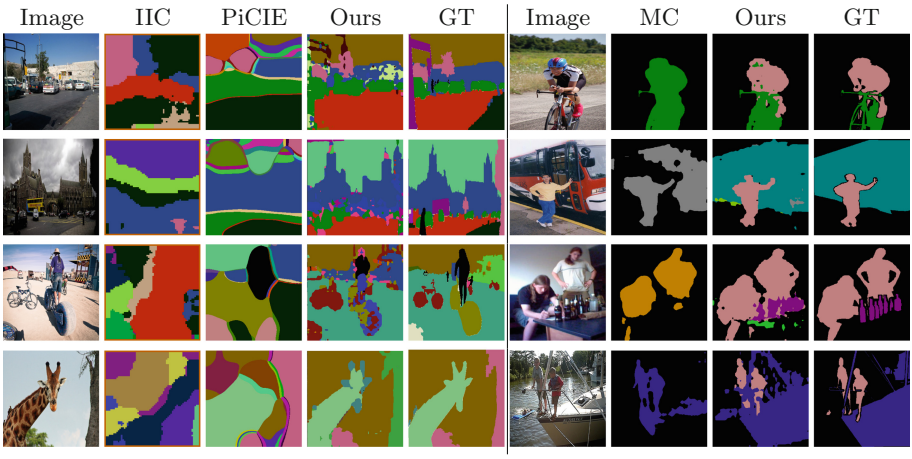

Fig. 3. Qualitative comparison on COCO-S-171 (left) and Pascal-VOC (right).

4.4 Ablation Studies

The Bootstrapping Mechanism. Table 3 compares results on MS-COCO of directly using initial pseudo labels ("initial"), training the student network once on the initial pseudo labels without the bootstrapping ("trained"), and the proposed bootstrapping mechanism ("bootstrapped"). Student network trained once on the initial pseudo labels can achieve an improved mIoU over the original pseudo labels, indicating that it can effectively learn meaningful information from the noisy pseudo label. The performance is further improved as more iterations of the teacher-student bootstrapping are introduced.

Table 3. Results for the effectiveness of bootstrap mechanism on MS-COCO with various semantic levels.

Dataset	Initial	Trained	Bootstrapped
COCO-S-27	8.95	13.19	16.19
COCO-S-171	5.05	8.66	11.93
COCO-80	4.23	8.28	12.69

Table 4. Results for the effectiveness of encoder and decoder on COCO-S-171.

Encoder	Decoder	mIoU	Acc.
None	None	5.05	17.33
R50+FPN	Segmenter	9.21	26.17
ViT+FPN	Classifier	7.96	24.17
R50+FPN	Classifier	8.33	25.35

Table 5. Results for different losses on COCO-S-171 and Pascal-VOC.

Dataset	Loss				mIoU	Acc.	Dataset	Loss				mIoU	Acc.
	CE	Peer	unc	div				CE	Peer	unc	div		
COCO-S-171	✓				10.49	29.72	Pascal-VOC	✓				34.24	79.85
		✓			10.54	30.46			✓			35.08	80.92
		✓	✓		11.24	33.03			✓	✓		36.46	82.36
		✓		✓	10.96	32.45			✓		✓	35.97	82.03
		✓	✓	✓	11.93	34.32			✓	✓	✓	37.15	83.59

The Effectiveness of Encoder and Decoder. To evaluate the effectiveness of the encoder and decoder used in our top-down pipeline, we conduct three more experiments that gradually replace the encoder and decoder with the ones used in PiCIE [10]. First, we change the encoder from ViT to ResNet-50 followed by an FPN model and keep the decoder as Segmenter. Second, we keep the encoder as ViT and change the decoder from Segmenter to a single layer convolution classifier to map the output dimension of the last attention block from d to K. Last, the encoder and decoder are replaced by ResNet-50 and convolution classifier, respectively. In all the settings, the encoders (ViT, ResNet-50) are trained on the ImageNet by DINO, and the decoders (Segmenter, convolution classifier) are randomly initialized. The results are shown in Table 4. As one can see, in all the settings, the performance can be improved from the initial pseudo label (encoder and decoder are all set to 'None'). Moreover, it's important to use fully-transformer architecture in our pipeline due to its robustness to noises.

The Bootstrapping Loss. We compare different combinations of using the original CE loss in Eq. (9), peer loss in Eq. (13), uncertainty loss in Eq. (14), and diversity loss in Eq. (15) in Table 5. Each of the proposed three bootstrapping losses can effectively improve the performance of the original CE loss, and their combination achieves the best performance.

5 Conclusion

We propose the first top-down framework for unsupervised semantic segmentation, which shows the importance of high-level semantic information in this

task. The semantic prior information is learned from large-scale visual data in a self-supervised manner, and then mapped to the pixel-level feature space. By carefully designing the mapping process and the unsupervised training mechanism, we obtain fine-grained segmentation for both foreground and background. Our design also enables the flexible control of granularity levels of the semantic categories, making it possible to generate semantic segmentation results for various datasets with different requirements. The fully unsupervised manner and the flexibility make our method much more practical in real applications.

Acknowledgements. This work was supported by funds for Key R&D Program of Hunan (2022SK2104), Leading plan for scientific and technological innovation of high-tech industries of Hunan (2022GK4010), the National Natural Science Foundation of Changsha (kq2202176), National Key R&D Program of China (2021YFF0900602), the National Natural Science Foundation of China (61672222) and Alibaba Group through Alibaba Research Intern Program.

References

1. Abdal, R., Zhu, P., Mitra, N., Wonka, P.: Labels4Free: unsupervised segmentation using StyleGAN. arXiv preprint arXiv:2103.14968 (2021)
2. Bielski, A., Favaro, P.: Emergence of object segmentation in perturbed generative models. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems, pp. 7256–7266 (2019)
3. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. Adv. Neural. Inf. Process. Syst. **33**, 9912–9924 (2020)
4. Caron, M., et al.: Emerging properties in self-supervised vision transformers. In: Proceedings of the International Conference on Computer Vision (ICCV) (2021)
5. Chefer, H., Gur, S., Wolf, L.: Transformer interpretability beyond attention visualization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 782–791 (2021)
6. Chen, M., Artières, T., Denoyer, L.: Unsupervised object segmentation by redrawing. In: Advances in Neural Information Processing Systems 32 (NIPS 2019), pp. 12705–12716. Curran Associates, Inc. (2019)
7. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607. PMLR (2020)
8. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020)
9. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15750–15758 (2021)
10. Cho, J.H., Mall, U., Bala, K., Hariharan, B.: PiCIE: unsupervised semantic segmentation using invariance and equivariance in clustering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16794–16804 (2021)
11. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3213–3223 (2016)

12. Dosovitskiy, A., et al.: An image is worth 16x16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
13. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vision **88**(2), 303–338 (2010)
14. Gong, K., Liang, X., Zhang, D., Shen, X., Lin, L.: Look into person: self-supervised structure-sensitive learning and a new benchmark for human parsing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 932–940 (2017)
15. Grill, J.B., et al.: Bootstrap your own latent: a new approach to self-supervised learning. In: Neural Information Processing Systems (2020)
16. Harb, R., Knöbelreiter, P.: InfoSeg: unsupervised semantic image segmentation with mutual information maximization (2021)
17. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9729–9738 (2020)
18. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969 (2017)
19. Hwang, J.J., et al.: SegSort: segmentation by discriminative sorting of segments. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 7334–7344 (2019)
20. Ji, X., Henriques, J.F., Vedaldi, A.: Invariant information clustering for unsupervised image classification and segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9865–9874 (2019)
21. Kanezaki, A.: Unsupervised image segmentation by backpropagation. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1543–1547. IEEE (2018)
22. Kim, D., Hong, B.W.: Unsupervised segmentation incorporating shape prior via generative adversarial networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 7324–7334 (2021)
23. Kim, W., Kanezaki, A., Tanaka, M.: Unsupervised learning of image segmentation based on differentiable feature clustering. IEEE Trans. Image Process. **29**, 8055–8068 (2020)
24. Li, C., et al.: Efficient self-supervised vision transformers for representation learning. arXiv preprint arXiv:2106.09785 (2021)
25. Li, X., et al.: Dense semantic contrast for self-supervised visual representation learning. arXiv preprint arXiv:2109.07756 (2021)
26. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
27. Liu, Y., Guo, H.: Peer loss functions: learning from noisy labels without knowing noise rates. In: International Conference on Machine Learning, pp. 6226–6236. PMLR (2020)
28. Mirsadeghi, S.E., Royat, A., Rezatofighi, H.: Unsupervised image segmentation by mutual information maximization and adversarial regularization. IEEE Robot. Autom. Lett. **6**(4), 6931–6938 (2021)
29. Ouali, Y., Hudelot, C., Tami, M.: Autoregressive unsupervised image segmentation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12352, pp. 142–158. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58571-6_9
30. Pinheiro, P.O., Almahairi, A., Benmalek, R.Y., Golemo, F., Courville, A.C.: Unsupervised learning of dense visual representations. In: NeurIPS (2020)

31. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017)
32. Shi, X., Khademi, S., Li, Y., van Gemert, J.: Zoom-cam: generating fine-grained pixel annotations from image labels. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 10289–10296. IEEE (2021)
33. Strudel, R., Garcia, R., Laptev, I., Schmid, C.: Segmenter: transformer for semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021)
34. Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Van Gool, L.: Unsupervised semantic segmentation by contrasting object mask proposals. In: International Conference on Computer Vision (2021)
35. Wang, X., Zhang, R., Shen, C., Kong, T., Li, L.: Dense contrastive learning for self-supervised visual pre-training. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3024–3033 (2021)
36. Wang, Z., et al.: Exploring set similarity for dense self-supervised representation learning. arXiv preprint arXiv:2107.08712 (2021)
37. Xie, Z., Lin, Y., Zhang, Z., Cao, Y., Lin, S., Hu, H.: Propagate yourself: exploring pixel-level consistency for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16684–16693 (2021)
38. Xu, J., Wang, X.: Rethinking self-supervised correspondence learning: a video frame-level similarity perspective. arXiv preprint arXiv:2103.17263 (2021)
39. Yao, Z., Cao, Y., Lin, Y., Liu, Z., Zhang, Z., Hu, H.: Leveraging batch normalization for vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 413–422 (2021)
40. Zou, Y., et al.: PseudoSeg: designing pseudo labels for semantic segmentation. In: International Conference on Learning Representations (ICLR) (2021)