MCIBI++: Soft Mining Contextual Information Beyond Image for Semantic Segmentation

Zhenchao Jin, Dongdong Yu, Zehuan Yuan, and Lequan Yu, Member, IEEE

Abstract—Co-occurrence visual pattern makes context aggregation become an essential paradigm for semantic segmentation. The existing studies focus on modeling the contexts within image while neglecting the valuable semantics of the corresponding category beyond image. To this end, we propose a novel soft mining contextual information beyond image paradigm named MCIBI++ to further boost the pixel-level representations. Specifically, we first set up a dynamically updated memory module to store the dataset-level distribution information of various categories and then leverage the information to yield the dataset-level category representations during network forward. After that, we generate a class probability distribution for each pixel representation and conduct the dataset-level context aggregation with the class probability distribution as weights. Finally, the original pixel representations are augmented with the aggregated dataset-level and the conventional image-level contextual information. Moreover, in the inference phase, we additionally design a coarse-to-fine iterative inference strategy to further boost the segmentation results. MCIBI++ can be effortlessly incorporated into the existing segmentation frameworks and bring consistent performance improvements. Also, MCIBI++ can be extended into the video semantic segmentation framework with considerable improvements over the baseline. Equipped with MCIBI++, we achieved the state-of-the-art performance on seven challenging image or video semantic segmentation benchmarks.

Index Terms—Semantic segmentation, context aggregation, dataset statistics, memory module, iterative testing, video semantic segmentation

1 INTRODUCTION

SEMANTIC segmentation is a long-standing challenging task in computer vision field, aiming to assign a semantic label to each pixel in an image. This task is of broad interest for potential applications of autonomous driving, medical diagnosing, robot sensing, to name a few. In recent years, deep neural networks [1], [2] have become the dominant solutions [3], [4], [5], [6], [7], [8], [9], [10], where the encoder-decoder architecture [11] is the cornerstone of these methods. Specifically, the current studies include adopting the graphical models or the cascade structure to refine the segmentation results [12], [13], [14], [15], [16], designing novel network backbones to extract more discriminative feature representations [17], [18], [19], [20], [21], replacing the conventional CNN-based decoder with the transformer-based decoder [22], aggregating the contextual information to augment the original pixel-level representations [4], [5], [9], [23], [24], [25], and so on.

As there exist co-occurrence visual patterns in images, modeling the context, i.e., relations of different pixels, is universal and essential for semantic segmentation. Prior to this work, the context of a pixel typically refers to a set of other pixels in the input image, e.g., the surrounding pixels. Specifically, PSPNet [4] utilizes the pyramid spatial pooling to aggregate the contextual information within the input image. DeepLab [5], [6], [7] exploits the contextual information by introducing the atrous spatial pyramid pooling (ASPP). OCRNet [26] proposes to improve the representation of each pixel by the weighted aggregation of the object region representations. Several recent works [8], [9], [27], [28] calculate the relations between a pixel and its contextual pixels and then aggregate the representations of the contextual pixels with higher weights for similar representations. Moreover, some studies [10], [29], [30] find that the pixel-wise cross entropy loss fundamentally lacks the spatial discrimination power and thereby propose to verify the segmentation structures directly with novel context-aware optimization objectives. Nonetheless, all of the existing approaches only model the context within individual images, while neglecting the valuable contextual information beyond the input image. The deep neural network-based semantic segmentation framework essentially categorizes the representations of pixels from the perspective of the whole dataset rather than the single input image in a non-linear embedding space. Therefore, to categorize a pixel accurately, the semantic information of the corresponding category in the other images should also be considered.

To overcome the aforementioned limitation of existing approaches, this paper proposes to mine the contextual information beyond the input image for semantic segmentation so that the pixel representations can be further improved. As
illustrated in Fig. 1, we first set up a memory module to store the dataset-level distribution information of various categories. Inspired by the network normalization techniques [31], [32], [33], [34], the distribution information in the memory module is dynamically updated according to the input images during the training while fixed in the inference phase. With the learned distribution information in the memory module, we further generate the dataset-level category representations to boost the original pixel representations. Specifically, we predict a class probability distribution for each pixel within the input image, where the distribution is learned under the supervision of the ground-truth segmentation. After that, we conduct the dataset-level context aggregation with the weighted aggregation of the yielded dataset-level category representations, where the weights are determined by the generated class probability distribution. Since the generated dataset-level representation contains the global statistic information of the corresponding category, it can incorporate more well-structured semantic information into the current input image to boost the segmentation performance from the perspective of the whole dataset.

Different from directly storing the dataset-level category representations, we propose to store the dataset-level distribution information of each category (i.e., mean and variances of category features) and utilize it to yield the demanded feature representations to reduce the memory usage during network training and boost the effectiveness of the dataset-level semantics. Thus, we refer our proposed paradigm as soft mining contextual information beyond image (MCIBI++). In addition, in our preliminary experiments as demonstrated in Table 1, we observe that employing the ground-truth segmentation to conduct dataset-level context aggregation (FCN+GT-MCIBI++) outperforms that with predicted class probability distribution (FCN+MCIBI++) with a large margin (i.e., 25.43% mIoU). This result indicates that the predicted class probability distribution of the pixel representations may be insufficient to effectively aggregate the dataset-level category representation. We thereby adopt a coarse-to-fine iterative inference strategy to progressively improve the accuracy of aggregation weights so as to incorporate more effective dataset-level semantics for each pixel.

Our MCIBI++ paradigm can be seamlessly incorporated into the existing segmentation frameworks (e.g., FCN [11], ISNet [23], PSPNet [4], UperNet [35] and DeepLabV3 [6]) and brings consistent yet solid performance improvements. Equipped with MCIBI++, our whole framework achieves the state-of-the-art intersection-over-union segmentation scores (mIoU) performance on six challenging benchmarks, i.e., ADE20K [36], LIP [37], Cityscapes [38], COCO-Stuff [39], PASCAL-Context [40] and PASCAL VOC 2012 [41], [42]. Furthermore, MCIBI++ can also be extended to the video semantic segmentation task, which needs to adopt the additional temporal information to obtain higher predictive accuracy. Extensive experiments on the VSPW [43] benchmark further demonstrate the effectiveness of our proposed paradigm. We expect this work to inspire more researches on dataset-level context aggregation for semantic segmentation.

In a nutshell, our main contributions are summarized as:

- This is one of the earliest works to explore how to mine the contextual information beyond the input image to further boost the segmentation performance. Specifically, we propose MCIBI++ paradigm to aggregate the dataset-level representations of various categories to boost the pixel-level representations of the current image.
- We design a simple yet effective memory module to store the dataset-level distribution information of various categories and thus yield more discriminative dataset-level category representation.
- We propose a novel dataset-level context aggregation scheme to incorporate the dataset-level semantic information to boost the discriminative capability of the original pixel-level representations.
- We propose an effective coarse-to-fine iterative inference strategy to progressively incorporate more accurate dataset-level category representations for each pixel.
- With the proposed MCIBI++ paradigm, we report the state-of-the-art performance on six image semantic segmentation benchmarks and one video semantic segmentation benchmark.

\section*{Difference From Conference Paper.} A preliminary version of this manuscript was previously published in [44]. In this expanded version, we significantly improve the conference version: (i) the design of the memory module, we replace the dataset-level category representations in the memory module with the dataset-level category distribution information and then employ the stored statistic information to yield the dataset-level category representations to reduce the memory size and improve the segmentation performance; (ii) the strategy in the inference phase, we further design a coarse-to-fine iterative inference strategy to progressively boost the segmentation performance through incorporating the more and more
accurate dataset-level category representations for the pixels. (iii) we conduct additional experiments and analysis on ADE20 K and Cityscapes datasets and compared with more recent methods to further show the state-of-the-art performance on six challenging benchmarks (i.e., ADE20 K, LIP, Cityscapes, COCO-Stuff, PASCAL-Context and PASCAL VOC 2012); (iv) we extend MCIBI++ to the video semantic segmentation task and also achieve consistent performance improvements and report the state-of-the-art results on the VSPW dataset. Due to the merits of mining contextual information beyond the input image as well as the generality and flexibility of our method, we obtain the second place in The 1st Video Scene Parsing in the Wild Challenge Workshop, ICCV2021 [43], [45] (the participants who won the first place used the extra training data from MS COCO [46]); (v) from both qualitative and quantitative perspective, our results are significantly better compared to our conference version [44].

2 RELATED WORK

Semantic Segmentation. The deep neural networks [1], [2] based methods have achieved remarkable success in semantic segmentation since the seminal work of fully convolutional networks (FCN) [11]. Modern semantic segmentation methods [3], [4], [5], [6], [7], [26] focus on further improving the framework of FCN. In details, to refine the coarse predictions of FCN, some researchers proposed to adopt the graphical models, such as CRF [13], [14], [47] and region growing [48]. CascadePSP [12] proposed to refine the segmentation results with a general cascade segmentation refinement model. Some novel backbone networks have also been designed to extract more effective feature representations [17], [18], [19]. Feature pyramid methods [4], [7], [49], [50], image pyramid methods [51], [52], [53], dilated convolutions [5], [6], [7] and context-aware optimization [10], [29], [30] are four popular paradigms to alleviate the limited receptive field issue or incorporate contextual information, in FCN. More recently, the academic attention has been geared toward transformer [54] and many efforts [20], [21], [22], [55], [56], [57] have been made to introduce the transformer architectures into image semantic segmentation. Specifically, SETR [21] proposed to improve the encoder-decoder based FCN architecture by encoding the input image as a sequence of patches. Swin Transformer [20] and SegFormer [56] proposed to construct the hierarchical features maps to enhance the pixel representations. MaskFormer [22] reformulated the semantic-level and instance-level segmentation as a mask classification task and achieved excellent results in both semantic and panoptic segmentation tasks. This work focuses on how to incorporate contextual information beyond image and the proposed paradigm can be utilized in both CNN-based and transformer-based segmentation approaches.

Context Scheme. Co-occurrent visual pattern makes aggregating long-range context in the final feature map become a common practice in semantic segmentation networks. To obtain the multi-scale contextual information, PSPNet [4] performs the spatial pooling at several grid scales, while DeepLab [5], [6], [7] proposed to adopt the pyramid dilated convolutions with different dilation rates. Based on DeepLab, DenseASPP [58] further densified the dilated rates to cover larger scale ranges. Recently, inspired by the self-attention scheme [54], many studies [3], [8], [9], [27], [59], [60] proposed to augment the pixel representations by a weighted aggregation of all pixel representations in the input image, where the weights are determined by the similarities between the pixel representations. Apart from this, some researches [26], [61], [62] proposed to first group the pixels into a set of regions and then improve the pixel representations by a weighted aggregation of these region representations, where the weights are the context relations between the pixel representations and the region representations. Despite impressive, all the existing approaches only exploit the contextual information within individual images. Different from these previous works, this paper proposes to mine more valuable contextual information for each pixel from the whole dataset to further improve the pixel representations. Concurrently, there are some very recent studies [63], [64], [65] that share the similar motivation with our work. Different from our method that aggregates the dataset-level contextual information beyond the input image via a memory module, these approaches extend the conventional contrastive learning strategy to the image semantic segmentation task to learn intra-class compact and inter-class dispersed features from the perspective of the whole dataset.

Memory Module. Memory module enables the deep neural networks to capture effective information beyond the current input image. It has shown power in various computer vision tasks [66], [67], [68], [69], [70], [71]. For instance, MEGA [67] adopted the memory mechanism to augment the candidate box features of key frame for video object detection, while MNE [69] leveraged the memory-based neighbourhood embedding to enhance the general convolutional features for image search and few shot learning tasks. For deep metric learning, XBM [66] designed a cross-batch memory module to mine informative examples across multiple mini-batches. Distinct from these previous methods, the memory module in this paper is used to store the dataset-level distribution information (i.e., mean and variance) of various categories rather than the feature representations themselves and the demanded dataset-level category representations will be generated according to the corresponding distribution information. In this case, we can capture the contextual information beyond the input image in a softer way to obtain more discriminative pixel-level representations from the perspective of the whole dataset. To the best of our knowledge, we are the first to utilize the idea of the memory module in supervised image semantic segmentation.

Video Semantic Segmentation. Video semantic segmentation aims to predict all pixels in a video with different semantic classes [43]. Previous methods [43], [72], [73], [74], [75], [76], [77], [78] can be divided into two streams. One stream is to explore the temporal relations of different frames via a warping module [72], [79] so that the segmentation accuracy can be improved. For example, DFF [79] proposed to adopt a flow field to propagate the deep feature maps from key frames to other frames. The other stream pays attention to improving the efficiency of the video semantic segmentation [73], [74], [75], [80]. For instance, ETC [80] proposed to leverage the knowledge distillation scheme to reduce the computing cost and Accel [73] proposed to combine the predictions of a reference branch to improve the segmentation.
efficiency. Recently, some researchers [43] introduced VSPW dataset, which is a large-scale video semantic segmentation benchmark. Based on the benchmark dataset, they proposed a generic temporal context blending scheme and achieved solid results on VSPW by extending OCRNet [26] and PSPNet [4] to the temporal dimension. In this paper, we aim to boost the segmentation accuracy by extending MCIBI++ into the video semantic segmentation task. Different from aggregating the pixel representations of some previous frames as the video context information [43], [72], [75], we exploit the temporal relations by aggregating the dataset-level representations of some previous frames to enhance the segmentation of the current one.

3 METHODOLOGY

In this section, we first introduce the paradigm of our soft mining contextual information beyond image (MCIBI++). Then, we describe the details of the proposed coarse-to-fine iterative inference strategy. Finally, we elaborate how to extend MCIBI++ to the video semantic segmentation task by adding the temporal information in the process of gathering the dataset-level representations for the pixels.

3.1 Overview of MCIBI++

Given an input RGB image $I \in \mathbb{R}^{3 \times H \times W}$, we first project the pixels in $I$ into a non-linear embedding space with a backbone network $F_B$ as follows:

$$R = F_B(I),$$

where the matrix $R$ with the size $Z \times \frac{H}{S} \times \frac{W}{T}$ denotes the pixel representations of $I$ and the feature dimension of the pixel representation is $Z$. As shown in Fig. 2, to mine the contextual information beyond the input image $C_{bi}$, we conduct the dataset-level context aggregation

$$C_{bi} = A_{bi}(H_{pr}(R), G(M)),$$

where $G$ is utilized to yield the dataset-level representations of various categories according to the dataset-level distribution information stored in the memory module $M$. $A_{bi}$ is the proposed dataset-level context aggregation scheme and $H_{pr}$ is a classification head used to predict the category probability distribution of the pixel representations in $R$. The matrix $C_{bi}$ with size of $Z \times \frac{H}{S} \times \frac{W}{T}$ denotes the dataset-level contextual information for each pixel.

To seamlessly integrate the proposed paradigm into the existing segmentation frameworks, we also define the self-existing context scheme within the individual images of the adopted framework as $A_{wi}$. Then we have

$$C_{wi} = A_{wi}(R),$$

where $C_{wi}$ is the contextual information within the input image. Next, the original pixel representation $R$ is augmented with both the proposed dataset-level contextual information and the individual image contextual information as follows:

$$R_{aug} = F_T(R, C_{bi}, C_{wi}),$$

where $F_T$ is a transform operator to fuse the original representations $R$, the contextual information beyond the input image $C_{bi}$ and the contextual information within the input image $C_{wi}$. Note that $C_{wi}$ is optional according to the adopted segmentation framework. For example, if the adopted segmentation framework is FCN without any contextual modules, $R_{aug}$ will be calculated with $F_T(R, C_{bi})$. However, if the used segmentation framework is DeepLab, $C_{wi}$ is the output of ASPP. Finally, the augmented pixel representation $R_{aug}$ is leveraged to generate the final semantic label of each pixel in the input image

$$O = UP_{8 	imes 1}(H_{cls}(R_{aug})), \quad (5)$$

where $H_{cls}$ is a classification head and $O$ is a matrix with size of $K \times H \times W$ and stores the semantic prediction of each pixel. $K$ is the number of the predicted classes.

3.2 Memory Module

As illustrated in Fig. 2b, the memory module $M$ with size of $K \times D$ is introduced to store the dataset-level distribution information of various classes, where $D$ denotes the number of hyperparameters used to represent the distribution. In this work, we model the dataset-level category representations as a Gaussian distribution and set $D = 2$ (i.e., mean and variance for each dataset-level category representation). During training, we first randomly pick one pixel representation for each class from the train split and then compute its mean and standard deviation to initialize the memory module $M$, where the representations are calculated by conducting $F_B(I)$. Then, the information in $M$ are updated by conducting moving average after each training iteration

$$M_t = (1 - m) \cdot M_{t-1} + m \cdot T(R_{t-1}), \quad (6)$$

where $t$ denotes for the current number of iterations and $m$ is the momentum. $T(R_{t-1})$ is used to transform the feature representation $R_{t-1}$ to the distribution information of each category. To obtain the distribution information from $R_{t-1}$, we first setup a matrix $M'$ with the same size of $M$, where the initial values in $M'$ are copied from $M$. For convenience of the presentation, we use the subscript $[i, j]$ or $[i, *]$ to index the single element or elements in a certain position or row in the matrix. $R$ is upsampled and permuted as size of $HW \times Z$, i.e., $R^{HW \times Z}$. Then, for each category $c_k$ existing in the image $I$, we extract the feature representations of corresponding pixels

$$R_{ck} = \{R_{[i, j]}^{HW \times Z} | (GT_{[i]} = c_k) \land (1 \leq i \leq HW)\}, \quad (7)$$

where $GT$ represents the ground truth category labels. After that, $R_{ck} \in \mathbb{R}^{N_{ck} \times C}$ stores the representations of category $c_k$, where $N_{ck}$ is the number of pixels labeled as $c_k$ in the input image $I$. Next, we calculate the average value of the pixel representations in $R_{ck}$ as follows:

$$\hat{R}_{ck} = \frac{1}{N_{ck}} \sum_{i=1}^{N_{ck}} R_{ck,[i, \ast]}.$$

Finally, the distribution information of $c_k$ in $M'$ is calculated as follows:
M = \sum_{i=1}^{Z} \frac{R_{c_k,i}}{Z}, \quad \mathcal{M}'_{c_k,0} = \sum_{i=1}^{Z} \frac{R_{c_k,i}}{Z}, \quad \mathcal{M}'_{c_k,1} = \sqrt{\frac{1}{Z} \sum_{i=1}^{Z} (R_{c_k,i} - \mathcal{M}'_{c_k,0})^2}. \tag{9}

M' will be updated with all the classes in the input image I and is regarded as the output of T to update the memory module M, following Eq. (6).

### 3.3 Dataset-Level Context Aggregation

Here, we elaborate the dataset-level context aggregation scheme \( \mathcal{A}_{dl} \) to adaptively aggregate the dataset-level category representations yielded by the dataset-level distribution information stored in \( \mathcal{M} \). As demonstrated in Fig. 2c, we first predict a weight matrix \( W \) with a size of \( K \times \frac{H}{s} \times \frac{W}{s} \) according to \( R \)

\[ W = \mathcal{H}_{pr}(R), \tag{10} \]

where \( W \) represents the category probability distribution of each pixel representation in \( R \). The classification head \( \mathcal{H}_{pr} \) is implemented with two \( 1 \times 1 \) convolutional layers following the Softmax function. Meanwhile, we generate the dataset-level representations of various categories \( \mathcal{C}_{dl} \) according to the distribution information in \( \mathcal{M} \)

\[ \mathcal{C}_{dl} = \mathcal{G}(\mathcal{M}), \tag{11} \]

where \( \mathcal{C}_{dl} \) is with the size of \( K \times Z \) and the generator \( \mathcal{G} \) is a sampling operation from the Gaussian distribution determined by the parameters in \( \mathcal{M} \). The pseudo code of \( \mathcal{G} \) is shown in Algorithm 1.

**Algorithm 1. Pseudo Code of Generator \( \mathcal{G} \)**

```python
# mean of each class means = memory_module[:, 0]
# standard deviation of each class stds = memory_module[:, 1]
# initialize dataset-level representations dl_representations = []

for cls_idx in range(K):
    dl_representation[cls_idx] = torch.normal(
        mean=torch.full((1, Z), means[cls_idx]),
        std=torch.full((1, Z), stds[cls_idx]))
```

After obtaining the dataset-level category representations \( \mathcal{C}_{dl} \) and the weight matrix \( W \), we calculate the coarse dataset-level representation for each pixel as follow:

\[ \hat{C}_{bi} = \text{permute}(W) \otimes \mathcal{C}_{dl}, \tag{12} \]

where \( \hat{C}_{bi} \) is with the size of \( \frac{HW}{s^2} \times Z \) and stores the aggregated dataset-level category representations for the pixel representations in \( R \). The \( \text{permute} \) operation is used to arrange \( W \) to the size of \( \frac{HW}{s^2} \times K \) and \( \otimes \) denotes the matrix multiplication. Since \( \mathcal{H}_{pr} \) only leverages the original pixel representation \( R \) to predict \( W \), while the pixel representations may be mis-classified. Therefore, we further calculate the relation between \( R \) and \( \hat{C}_{bi} \) to obtain a position confidence weight \( P \) to further
recalibrate $\bar{C}_b$ as follows:

$$P = \text{Softmax} \left( \frac{\mathcal{F}_{R_1}(\text{permute}(R)) \otimes \mathcal{F}_{R_2}(\bar{C}_b)^T}{\sqrt{C}} \right),$$

$$C_b = \text{permute}(\mathcal{F}_O(P \otimes \mathcal{F}_C(\bar{C}_b))),$$

(13)

where the $\text{permute}(R)$ operation is adopted to arrange $R$ to the size of $\frac{H}{8} \times Z$. $\mathcal{F}_{R_1}$, $\mathcal{F}_{R_2}$, $\mathcal{F}_O$ and $\mathcal{F}_C$ are introduced to adjust the dimension of each pixel representation and they are implemented by a $1 \times 1$ convolutional layer. The last $\text{permute}$ operation is used to resize the output to have the size of $\frac{H}{8} \times \frac{W}{8}$.

### 3.4 Objective Functions

A joint loss function of the weight matrix $W$ and the final prediction $O$ is used to optimize the model parameters. Specifically, the objective of $W$ is defined as

$$\mathcal{L}_W = \frac{1}{H \times W} \sum_{i,j} CE(W'_{s[i,j]}, GT_{ij}),$$

(14)

where $W'$ is the upsampled prediction $UP_{8 \times 8}(W)$. $CE$ denotes for the cross entropy loss and $\sum_{i,j}$ denotes that the summation is calculated over all locations on the input image $I$. Moreover, to make the final segmentation prediction $O$ contain the accurate category probability distribution of each pixel, we define the objective of $O$ as

$$\mathcal{L}_O = \frac{1}{H \times W} \sum_{i,j} CE(O_{s[i,j]}, GT_{ij}).$$

(15)

Finally, we formulate the joint loss function $\mathcal{L}$ to train the whole segmentation framework as follows:

$$\mathcal{L} = \alpha \mathcal{L}_W + \mathcal{L}_O,$$

(16)

where $\alpha$ is the hyper-parameter to balance the two losses. Following [44], we empirically set $\alpha = 0.4$ by default. With this joint loss function, the model parameters are updated jointly through back propagation. Note that the values in $M$ are not learnable through the back propagation.

### 3.5 Coarse-to-Fine Iterative Inference Strategy

In the inference phase, we develop a coarse-to-fine iterative inference strategy to progressively incorporate accurate dataset-level representation for each pixel. As indicated in Fig. 3, in Stage1, we first conduct Eqs. (10), (11), and (12) to gather the first-stage dataset-level representations $C_{b1}$ for the pixels. We employ $C_{b1}$ to augment the original pixel representations $R$ and obtain the first-stage category probability distribution of each pixel, i.e., $O_{s1}$. During Stage2, we employ the first-stage prediction $O_{s1}$ to update the weights for dataset-level context aggregation. Specifically, we calculate the weights in Stage2 as

$$W_{s2} = \frac{O_{s1} + W}{2}.$$  

(17)

After that, we replace $W_{s1}$ (i.e., $W$ predicted by Eq. (10)) with $W_{s2}$ and re-conduct the dataset-level context aggregation as Eqs. (10), (11), and (12) to obtain $C_{b2}$. Finally, we utilize the pixel representations augmented by $\bar{C}_{b2}$ to predict the final class probability distribution of each pixel, i.e., $O_{s2}$. As the predicted category probability distribution in $O_{s1}$ is more accurate than $W$, $W_{s2}$ can help to improve the accuracy of the aggregated dataset-level category representations for the pixels and thereby achieve better segmentation results.

Following the steps above, we can also obtain $W_{s3}$, $i \geq 2$ to further boost the segmentation performance. In particular, we set $i = 2$ by default for best trade-off.

### 3.6 Extension to Video Semantic Segmentation

In this subsection, we extend MCIBI++ to the video semantic segmentation task. The key idea of the extension is to introduce the extra temporal information by leveraging the memory module. Accordingly, the image semantic segmentation framework equipped with MCIBI++ can be effectively and conveniently extent to the temporal dimension.

Baseline Framework. Although video semantic segmentation is different from image semantic segmentation, a stronger image semantic segmentor can usually achieve better performance in the video semantic segmentation. Therefore, we adopt the image semantic segmentation framework equipped with MCIBI++ as our baseline framework. Specifically, we first perform Eqs. (1), (2), and (3) to obtain $R$, $C_b$, and $C_{w1}$. For our implementation, the adopted backbone networks are ResNet-101 and Swin-Large [20] and the employed context module within image is PPM plus FPN proposed in UperNet [35].

Incorporating Temporal Information. As demonstrated in Fig. 4, we also feed the historical frames into the network to introduce the temporal information for our video semantic segmentation framework. In details, given the historical frame $I_{N-i}$, $1 \leq i \leq h$, we first adopt the backbone network to extract the historical pixel representations $R_{N-i}$. Then, we calculate the aggregation weights for these historical pixel representations as follow:

$$W_{N-i} = \mathcal{H}_p(R_{N-i}), \quad 1 \leq i \leq h,$$

(18)

where $h$ is the number of historical frames. In our implementation, we set $h = 2$. Then, we leverage the calculated aggregation weights $W_{N-i}$ to gather the dataset-level
representations for each historical frame by performing Eqs. (11), (12), and (13) and thereby we can obtain \( C_{bi}^{N-i}, 1 \leq i \leq h \). Finally, we combine them together to obtain the augmented pixel representation

\[
R_{aug} = F_T(R, C_{wi}, C_{bi}, C_{bi}^{N-1}, \ldots, C_{bi}^{N-h})
\]

where \( R_{aug} \) is as the input of Eq. (5) to predict the category probability distribution of each pixel in \( I_N \).

**Discussion.** In fact, there are other strategies to exploit the temporal information from the historical frames. For example, \( C_{bi}^{N-i} \) in Eq. (19) can also be replaced with \( R_{N-i} \), which seems more convenient to be implemented. However, according to our experience, introducing \( C_{wi} \) and \( C_{bi} \) to \( R_{aug} \) can bring more performance gains for video semantic segmentation than the direct temporal information \( R_{N-i} \). If we adopt \( R_{N-i} \) in Eq. (19), the proportion of the contextual information will decrease with the increase of the utilized historical frames, which may suppress the segmentation performance. In addition, the appearance of the same object may vary a lot in different frames and thus the pixel representations in \( R_{N-i} \) for the same object may also vary a lot during training, which is not friendly for the network to capture temporal relations of the objects between the frames. This problem can be alleviated when adopting \( C_{bi}^{N-i} \), as the dataset-level representation for each object keeps consistent during different frames.

### 4 EXPERIMENTS ON IMAGE SEMANTIC SEGMENTATION

We first present an extensive evaluation of our framework on the image semantic segmentation task.

#### 4.1 Experimental Setup

**Datasets.** We conduct experiments on four widely-used semantic segmentation benchmarks:

- **ADE20 K** [36] is a challenging scene parsing dataset that contains 150 classes and diverse scenes with 1,038 image-level labels. The training, validation, and test sets consist of 20 K, 2 K, 3 K images, respectively.
- **LIP** [37] is tasked for single human parsing. There are 19 semantic classes and one background class. The dataset is divided into 30 K/10 K/10 K images for training, validation and testing, respectively.
- **Cityscapes** [38] densely annotates 19 object categories in urban scene understanding images. This dataset consists of 5,000 finely annotated images, where 2,975 are available for training, 500 for validation and 1,524 for testing.
- **COCO-Stuff** [39] is a challenging scene parsing dataset that provides rich annotations for 91 thing classes and 91 stuff classes. This work adopts the old version which includes 10 K images (i.e., 9 K for training and 1 K for testing) from MS COCO [46].

**Training Settings.** Our backbone network was initialized by the weights pre-trained on ImageNet [81], while the remaining layers were randomly initialized. Following the previous work [5], the learning rate was decayed according to the “poly” learning rate policy with factor \((1 - \frac{iter}{iter_{max}})^{0.9}\). Synchronized batch normalization implemented by pytorch was enabled during training. For data augmentation, we adopted random scaling, horizontal flipping and color jitter. More specifically, the training settings for different datasets are listed as follows:

- **ADE20 K:** We set the initial learning rate as 0.01, weight decay as 0.0005, crop size as 512 × 512, batch size as 16 and training epochs as 130 by default.
- **LIP:** The initial learning rate was set as 0.01 and the weight decay was 0.0005. We set the crop size of the input image as 473 × 473 and batch size as 32 by default. Besides, if not specified, the networks were fine-tuned for 150 epochs on the train set.
- **Cityscapes:** The initial learning rate was set as 0.01 and the weight decay was 0.0005. We randomly cropped out high-resolution patches 512 × 1024 from the original images as the inputs for training. The batch size and training epochs were set as 8 and 220, respectively.
- **COCO-Stuff:** We set the initial learning rate as 0.001, weight decay as 0.0001, crop size as 512 × 512, batch size as 16 and training epochs as 110 by default.

**Inference Settings.** For ADE20 K, COCO-Stuff and LIP, the size of the input image during testing was the same as the size of the input image during training. And for Cityscapes, the input image was resized to have its shorter side being 1,024 pixels. By default, no tricks (e.g., multi-scale with flipping testing) were used during testing.
TABLE 2
Ablation Study of $F_T$ on ADE20 K Val Split

| Method              | Backbone | $F_T$ | mIoU (%) |
|---------------------|----------|-------|----------|
| FCN                 | ResNet-50|       | 36.96    |
| FCN+MCIBI++ (ours)  | ResNet-50| add   | 41.15    |
| FCN+MCIBI++ (ours)  | ResNet-50| weighted add | 42.06 |
| FCN+MCIBI++ (ours)  | ResNet-50| concatenation | 43.39 |

Evaluation Metrics. Following the standard setting, mean intersection-over-union (mIoU) was adopted for evaluation.

Reproducibility. Our method was implemented with PyTorch (version $\geq 1.3$) [82] and was trained on eight NVIDIA Tesla V100 GPUs with a 32 GB memory per-card. All the testing procedures were performed on a single NVIDIA Tesla V100 GPU. To provide full details of our framework, our code will be made publicly available in [83].

4.2 Ablation Analysis

Transform Function $F_T$. Table 2 shows the performance of three fusion strategies to fuse $R$ and $C_b$ on the validation set of ADE20 K. Specifically, add denotes for element-wise adding, which brings 4.19% mIoU improvements compared with the baseline model FCN. weighted add indicates weighted summation of $R$ and $C_b$, where the weights are predicted by a $1 \times 1$ convolutional layer. It achieves 5.10% mIoU improvements over the baseline model and outperforms the naive summation. concatenation stands for concatenating $R$ and $C_b$ and achieves the optimal performance among three strategies. All the three strategies achieve the consistent improvements over the baseline model, indicating that the proposed MCIBI++ paradigm can effectively improve the pixel representations so that the networks can classify the pixels more accurately.

Memory Module. Table 3 shows the ablation study of the memory module $M$ on the validation set of ADE20 K. Comparing the 1st and 4th rows, we can observe that leveraging $M$ to introduce the contextual information beyond the input image can boost the performance of baseline segmentation framework (from 36.96% to 43.39% mIoU). This result indicates that the yielded dataset-level representations can effectively improve the original pixel representations. In addition, it is also observed that MCIBI++ (4th row) outperforms our previous MCIBI [44] (3 rd row) by 1.21%, showing that storing the dataset-level distribution information of each category instead of directly storing the feature representations can obtain more discriminative dataset-level category representations. Furthermore, the segmentation framework with non-updated $M$ achieves poor performance (2nd row, only hitting 21.78% mIoU), indicating that one key success factor of $M$ is the dynamic update mechanism to maintain the suitable dataset-level distribution information of various classes in the memory module.

We further conduct the ablation studies in Table 4 to find the most effective value of momentum $m$. It is observed that setting $m = 0.1$ achieve the optimal performance. Moreover, although adopting the polynomial annealing policy to schedule $m$ seems more flexible and has been adopted in [44], we observe that using a fixed momentum value brings the better segmentation results after replacing the dataset-level category representations with the dataset-level distribution information in $M$.

Loss Weight. Table 5 shows the ablation study of the loss weight $\alpha$ on the validation set of ADE20 K. We can see that setting $\alpha = 0.4$ achieve the best segmentation performance.

Coarse-to-Fine Iterative Inference Strategy. In our preliminary experiments as shown in Table 1, we find that improving the accuracy of aggregation weights $W$ can boost the segmentation performance. Therefore, we propose a coarse-to-fine iterative inference strategy to improve the segmentation results in the inference phase. From Table 3, we can observe that the designed iterative inference strategy brings 0.35% improvements on mIoU (comparing 4th and 5th rows), demonstrating the effectiveness of our strategy. To further validate this strategy, we also calculate the mIoU and accuracy of the aggregation weights $W$ at different stages. Particularly, $W_{i=2}$ obtains 40.89% mIoU and 79.40% pixel accuracy while $W_{i=1}$ only achieves 36.44% mIoU and 77.03% pixel accuracy, which is along the lines of our motivation.

We further conduct ablation experiments in Table 6 to investigate the most effective number of the stages in the coarse-to-fine iterative inference strategy. From this table, it is observed that setting stage $i = 2$ can obtain the better trade-off between accuracy improvement and computation cost. Moreover, the resulting best model is very efficient and can be deployed in the real-world scenarios. We will make our code and data publicly available.

TABLE 3
Ablation Study of MCIBI++ on ADE20 K Val Split

| FCN Memory-R | Memory-D | Update | IIS | mIoU (%) |
|--------------|----------|--------|-----|---------|
| ✓            | ✓        | ✓      | ✓   | 36.96   |
| ✓            | ✓        |        | ✓   | 21.78   |
| ✓            | ✓        | ✓      | ✓   | 42.18   |
| ✓            | ✓        | ✓      | ✓   | 43.39   |
| ✓            | ✓        | ✓      | ✓   | 43.74   |

Memory-R means directly storing the dataset-level representations of categories in $M$, while Memory-D denotes maintaining the distribution information of classes in $M$. Update indicates whether to update the memory module during training and IIS refers to the proposed iterative inference strategy.

TABLE 4
Ablation Study of Momentum $m$ of $M$ on ADE20 K Val Set

| Momentum $m$ | Backbone | Stride | mIoU (%) |
|--------------|----------|--------|---------|
| 0.01         | ResNet-50| 8×     | 42.06   |
| 0.05         | ResNet-50| 8×     | 42.89   |
| 0.1          | ResNet-50| 8×     | 43.39   |
| 0.3          | ResNet-50| 8×     | 42.71   |
| 0.5          | ResNet-50| 8×     | 42.72   |
| 0.7          | ResNet-50| 8×     | 43.23   |
| 0.9          | ResNet-50| 8×     | 42.95   |
| 0.9-Poly     | ResNet-50| 8×     | 42.45   |

0.9-Poly means adopting the polynomial annealing policy to schedule $m$ [44].

TABLE 5
Ablation Study of Loss Weight $\alpha$ on ADE20 K Val Set

| Loss Weight $\alpha$ | Backbone | Stride | mIoU (%) |
|----------------------|----------|--------|---------|
| 0.2                  | ResNet-50| 8×     | 43.35   |
| 0.4                  | ResNet-50| 8×     | 43.39   |
| 0.6                  | ResNet-50| 8×     | 43.25   |
| 0.8                  | ResNet-50| 8×     | 42.83   |
| 1.0                  | ResNet-50| 8×     | 42.22   |
cost. (Note that stage \( i = 1 \) denotes for the baseline method without IIS.)

The Initialization of \( M \). By default, each item in \( M \) is initialized by the mean and std of a randomly selecting pixel representation of the corresponding category. To test the influence of the initialization method on the segmentation performance, we also initialize \( M \) with the mean and std of the composite vector of each category, where the composite vectors are obtained by cosine similarity clustering. The performance of this strategy is comparable with the random initialization (43.43\% versus 43.39\%), which is consistent with the observation in MCIBI [44]. This comparison indicates that the dynamic updating mechanism can make \( M \) converge and without relying on a strict initialization, which shows the robustness of our method.

Complexity Comparison. In Table 7, we show the efficiency of the proposed dataset-level context aggregation (DCA) scheme. Compared with the existing context schemes, our DCA requires the least computation complexity (242.80 GFLOPs) and the least parameters (14.82 M), showing that the proposed context module is light. Since DCA aims to aggregate the contextual information beyond the input image while the existing context schemes focus on gathering contextual information within the image, DCA is complementary to the existing context schemes. The lightweight attribute of DCA makes the whole model complexity still tolerable after integrating the proposed module with other existing within image context schemes to further boost performance. For example, after integrating DCA with ASPP, the parameters, FLOPs and inference time just increase by 3.42 M, 56.09 G and 11.20 ms, respectively (comparing the first row and the last row in Table 7), while the segmentation performance boosts from 43.19\% to 44.85\%. This comparison well validate our motivation, i.e., the contexts beyond image \( C_{bi} \) and the contexts within image \( C_{wi} \) are complementary. It is worth noting that only incorporating the MCIBI++ paradigm can also boost the baseline FCN framework to reach a 43.39\% mIoU, which is higher than most of the existing context schemes.

Training Memory. Table 7 also compares the maximum training memory required for MCIBI and MCIBI++. It is observed that the training memory decreases from 3.65 G to 2.96 G after replacing representations with distribution information in the memory module. The drop is mainly due to the reduction of the memory required to update the memory module \( M \). Specifically, during training, MCIBI leverages the feature representations stored in GPU to update \( M \) while MCIBI++ only utilizes the distribution information to update \( M \). Obviously, the dimension of the tensor storing distribution information is much smaller than that storing feature representations. Therefore, the proposed MCIBI++ can benefit in the saving of GPU memory during network training compared with our previous MCIBI. Note that, in the inference phase, the GPU memory requirements for MCIBI and MCIBI++ will decrease a lot since the memory module on GPU is without updating.

Overfitting. As the memory module first stores the dataset-level distribution information of different categories and then generate the dataset-level contextual information to augment the original pixel representations, this memory module may lead the segmentation framework to overfit on the current dataset. To investigate this point, we train the baseline segmentors (i.e., FCN and UperNet) and our proposed framework on the train set of LIP, and then evaluate them on the validation set of both LIP and CIHP [87], where CIHP is a much more challenging multi-human parsing dataset with the same categories as LIP and can be used to evaluate the generalization ability of the networks. As demonstrated in Table 8, our framework consistently boosts the segmentation performance of two baseline segmentors on both LIP val set (51.13\% versus 48.63\% and 53.92\% versus 48.63\%.)

| Method                     | LIP train set | LIP val set | mIoU (%) |
|----------------------------|---------------|-------------|-----------|
| FCN (our impl.)            | LIP train set | LIP val set | 48.63     |
| UperNet (our impl.)        | LIP train set | LIP val set | 52.95     |
| FCN+MCIBI++ (ours)         | LIP train set | LIP val set | 51.13 (+2.50) |
| UperNet+MCIBI++ (ours)     | LIP train set | LIP val set | 53.92 (+0.97) |
| FCN+MCIBI++ (ours)         | LIP train set | CIHP val set | 27.57 (+0.37) |
| UperNet+MCIBI++ (ours)     | LIP train set | CIHP val set | 27.74 (+0.90) |

LIP and CIHP contain exactly the same semantic labels.
52.95% for FCN and Upernet, respectively) and CIHP val set (27.57% versus 27.20% and 27.74% versus 26.84%). These comparisons indicate that the distribution information of LIP also benefits the segmentation on CIHP dataset and introducing the memory module can help the segmentors improve the generalization ability to some extent.

### 4.3 Integrated Into Various Segmentation Frameworks

Our MCIBI++ is orthogonal to the segmentation network design and it can be seamlessly incorporated into existing segmentation frameworks. Therefore, we integrate the proposed MCIBI++ into five existing segmentation frameworks, i.e., FCN, PSPNet, Upernet, DeepLabV3 and ISNet to further validate the effectiveness and robustness of our approach. Table 9 shows the performance comparison on four benchmark datasets, i.e., ADE20 K, Cityscapes, LIP and COCO-Stuff. For a fair comparison, all the results are obtained under the single-scale testing without using the proposed coarse-to-fine iterative inference strategy. It is observed that MCIBI++ consistently improves the segmentation performance of the four baseline frameworks by large margins on all datasets. For example, for the segmentation framework without within image context module A_{wi} (i.e., FCN), MCIBI++ brings 6.43% mIoU improvements on ADE20 K, 3.61% mIoU improvements on Cityscapes, 2.50% mIoU improvements on LIP and 5.62% mIoU improvements on COCO-Stuff, respectively. For a stronger baseline with A_{wi} (e.g., DeepLabV3), MCIBI++ boosts the mIoU performance from 43.19% to 44.85% on ADE20 K, 79.62% to 80.72% on Cityscapes, 52.35% to 53.59% on LIP and 37.63% to 38.94% on COCO-Stuff, respectively. These comparison demonstrates the flexibility and generalizability of our proposed MCIBI++ paradigm.

### 4.4 Comparison With the State-of-the-Art Methods

#### Results on ADE20K

Table 10 compares the segmentation results on the validation set of ADE20 K. With the ResNet-101 backbone, ISNet [23] achieves 47.31% mIoU via integrating the image-level and semantic-level context for semantic segmentation, which is the previous best method. Based on the same ResNet-101 backbone, our approach achieves superior mIoU of 47.93% with a significant margin over ISNet. Moreover, our Upernet+MCIBI++ with the ResNet-101 backbone even outperforms the previous ISNet, DeepLabV3 and DeepLabV3+MCIBI with a more powerful ResNeSt-101 [18] backbone by 0.38%, 1.02% and 0.57%, respectively. Without transformer-based backbone, the state-of-the-art segmentation framework is DeepLabV3 with ResNeSt-200, hitting 48.36% mIoU. Due to the superiority of the proposed MCIBI++, our Upernet+MCIBI++ with a relatively light backbone network ResNeSt-101 still reports the new state-of-the-art performance (i.e., 48.56% mIoU) among the transformer-free methods. In addition, prior to this work, Upernet achieves the state-of-the-art with 53.50% mIoU by adopting a transformer-based backbone network, i.e., Swin-Large [20]. Based on this, our method Upernet+MCIBI++ with the same Swin-Large backbone achieves a new state-of-the-art result with

#### Table 9
The Segmentation Performance Improvements on Various Benchmarks When Integrating MCIBI++ Into Four Existing Segmentation Frameworks

| Method (our impl.) | Backbone | Stride | ADE20 K (train/val) | Cityscapes (train/val) | LIP (train/val) | COCO-Stuff (train/val) |
|--------------------|----------|--------|---------------------|------------------------|----------------|------------------------|
| FCN (ours)         | ResNet-50 | 8x     | 43.39 (+6.43)       | 78.77 (+3.61)          | 51.13 (+2.50)  | 37.38 (+5.62)          |
| PSPNet+MCIBI++     | ResNet-50 | 8x     | 43.89 (+1.25)       | 79.91 (+0.86)          | 52.93 (+0.99)  | 38.47 (+1.07)          |
| Upernet+MCIBI++    | ResNet-50 | 8x     | 44.30 (+1.28)       | 80.05 (+0.97)          | 53.92 (+0.97)  | 39.20 (+1.55)          |
| DeepLabV3+MCIBI++ | ResNet-50 | 8x     | 44.85 (+1.66)       | 80.72 (+1.10)          | 53.59 (+1.24)  | 38.94 (+1.31)          |
| ISNet              | ResNet-50 | 8x     | 45.33 (+1.11)       | 80.49 (+1.17)          | 54.06 (+0.92)  | 39.33 (+1.27)          |

#### Table 10
The Comparison With the State-of-the-Art Methods on ADE20 K (Val)

| Method                  | Backbone     | Stride | mIoU (%) |
|-------------------------|--------------|--------|----------|
| CCNet [28]              | ResNet-101   | 8x     | 45.76    |
| OCNet [27]              | ResNet-101   | 8x     | 45.45    |
| ACNet [24]              | ResNet-101   | 8x     | 45.90    |
| DMNet [84]              | ResNet-101   | 8x     | 45.50    |
| EncNet [85]             | ResNet-101   | 8x     | 44.65    |
| PSPNet [4]              | ResNet-101   | 8x     | 43.29    |
| PSANet [59]             | ResNet-101   | 8x     | 43.77    |
| APCNet [86]             | ResNet-101   | 8x     | 45.38    |
| OCRNet [26]             | ResNet-101   | 8x     | 45.28    |
| Upernet [35]            | ResNet-101   | 8x     | 44.85    |
| ISNet [23]              | ResNet-101   | 8x     | 47.31    |
| MaskFormer [22]         | ResNet-101   | 8x     | 47.20    |
| PSPNet [4]              | ResNet-269   | 8x     | 44.94    |
| OCRNet [26]             | HRNetV2-W48   | 4x     | 45.66    |
| ISNet [23]              | ResNet-101   | 8x     | 47.55    |
| DeepLabV3 [6]           | ResNet-101   | 8x     | 46.91    |
| DeepLabV3 [6]           | ResNet-269   | 8x     | 48.36    |
| SETR-MLA [21]           | ViT-Large    | 16x    | 50.28    |
| Upernet [20]            | Swin-Large   | 32x    | 53.50    |
| DeepLabV3+MCIBI++ [44]  | ResNet-101   | 8x     | 47.72    |
| Upernet+MCIBI++ [ours]  | ResNet-101   | 8x     | 47.36    |
| DeepLabV3+MCIBI++ [ours]| ResNet-101   | 8x     | 48.55    |
| Upernet+MCIBI++ [ours]  | ResNet-101   | 8x     | 48.56    |
| Upernet+MCIBI++ [ours]  | ResNet-101   | 8x     | 48.56    |

Multi-scale and flipping testing are adopted for fair comparison. Bold denotes the best score.
TABLE 11
Segmentation Results on the Validation Set of LIP

| Method                  | Backbone       | Stride | mIoU (%) |
|-------------------------|----------------|--------|----------|
| Attention [51]          | ResNet-101     | -      | 42.92    |
| DeepLab [5]             | ResNet-101     | -      | 44.80    |
| MMAN [88]               | ResNet-101     | -      | 46.81    |
| CE2P [16]               | ResNet-101     | 16×    | 53.10    |
| BraidNet [89]           | ResNet-101     | 8×     | 54.54    |
| DeepLabV3 [6]           | ResNet-101     | 8×     | 54.58    |
| OCNet [27]              | ResNet-101     | 8×     | 54.72    |
| CCNet [28]              | ResNet-101     | 8×     | 55.47    |
| OCRNet [26]             | ResNet-101     | 8×     | 55.60    |
| CorrPM [90]             | ResNet-101     | 8×     | 55.33    |
| ISNet [23]              | ResNet-101     | 8×     | 55.41    |
| ISNet [23]              | ResNet-St-101  | 8×     | 56.81    |
| HRNet [19]              | HRNetV2-W48    | 4×     | 55.90    |
| OCRNet [26]             | HRNetV2-W48    | 4×     | 56.65    |
| DeepLabV3+MCIBI++ ([44] | ResNet-101     | 8×     | 55.42    |
| DeepLabV3+MCIBI++ ([44] | ResNet-St-101  | 8×     | 56.34    |
| DeepLabV3+MCIBI++ ([44] | HRNetV2-W48    | 4×     | 56.99    |
| UperNet+MCIBI++ (ours)  | ResNet-101     | 8×     | 56.32    |
| UperNet+MCIBI++ (ours)  | ResNet-St-101  | 8×     | 57.08    |
| UperNet+MCIBI++ (ours)  | Swin-Large     | 32×    | 59.91    |

*Test time augmentation is utilized for fair comparison.*

mIoU hitting 54.52%. It is worth noting that ADE20 K is very challenging due to its various image sizes, the large number of semantic categories and the gap between the training and validation sets. Our framework achieves consistent improvements over counterparts and new state-of-the-art performance on this dataset, which clearly demonstrates the significance of our proposed MCIBI++ paradigm.

Results on LIP. Table 11 shows the comparison results on the validation set of LIP. If only considering the ResNet-101 backbone, we observe that UperNet+MCIBI++ yields an mIoU of 56.32%, which is 0.72% higher than the previous best segmentation framework OCRNet. Meanwhile, it can be observed that our UperNet-MCIBI++ with the ResNet-St-101 backbone achieves 57.08% mIoU, on par with state-of-the-art approaches. To further show the effectiveness of our method, we adopt UperNet+MCIBI++ with the transformer-based network Swin-Large as backbone and report the new state-of-the-art performance, hitting 59.91% mIoU. This is particularly impressive considering that human parsing is a more challenging fine-grained semantic segmentation task.

Results on COCO-Stuff. Table 12 reports the performance comparison of our method against nine competitors on the test set of COCO-Stuff. In the case of employing ResNet-101 backbone network, UperNet+MCIBI++ reaches 41.84% mIoU, outperforming the state-of-the-art ISNet by 0.24% mIoU. It is noteworthy that this performance is also superior to some methods with a stronger backbone, e.g., OCRNet with HRNetV2-W48 (41.84% versus 40.50%). With a more robust feature extractor ResNet-St-101, our method achieves mIoU of 42.71%, which surpasses the previous CNN-based best result (i.e., DeepLabV3+MCIBI) by 0.56% mIoU. Since recent academic attention has been geared toward transformer-based backbone networks, in Table 12, we have also integrated our method into the popular Swin-Large backbone. As we can see, UperNet+MCIBI++ with Swin-Large obtains 50.27% mIoU, which outperforms other competitors by a large margin. These results firmly suggest the superiority of the proposed soft mining the contextual information beyond the input image paradigm.

Results on Cityscapes. Table 13 shows the comparison results on the test set of Cityscapes. For a fair comparison, only the frameworks trained on the Cityscapes trainval set are added in this table. As observed, we first utilize the common ResNet-101 backbone network to train MCIBI++ and obtain 82.20% mIoU, which surpasses the previous best method with the same backbone by 0.17% mIoU (82.03% versus 82.20%). Note that, 82.03% mIoU for Cityscapes under the comparable settings (e.g., the

### TABLE 12
Comparison With the State-of-the-Art Segmentation Frameworks on the Test Set of COCO-Stuff

| Method                  | Backbone       | Stride | mIoU (%) |
|-------------------------|----------------|--------|----------|
| DANet [8]               | ResNet-101     | 8×     | 39.70    |
| OCRNet [26]             | ResNet-101     | 8×     | 39.50    |
| SVCNet [91]             | ResNet-101     | 8×     | 39.60    |
| EMA Net [62]            | ResNet-101     | 8×     | 39.90    |
| ACNet [24]              | ResNet-101     | 8×     | 40.10    |
| ISNet [23]              | ResNet-101     | 8×     | 41.60    |
| MaskFormer [22]         | ResNet-101     | 8×     | 39.30    |
| ISNet [23]              | ResNet-St-101  | 8×     | 42.08    |
| OCRNet [26]             | HRNetV2-W48    | 4×     | 40.50    |
| DeepLabV3+MCIBI++ ([44]| ResNet-101     | 8×     | 41.49    |
| DeepLabV3+MCIBI++ ([44]| ResNet-St-101  | 8×     | 42.15    |
| DeepLabV3+MCIBI++ ([44]| ViT-Large      | 16×    | 44.89    |
| UperNet+MCIBI++ (ours)  | ResNet-101     | 8×     | 41.84    |
| UperNet+MCIBI++ (ours)  | ResNet-St-101  | 8×     | 42.71    |
| UperNet+MCIBI++ (ours)  | Swin-Large     | 32×    | 50.27    |

*Multi-scale and flipping testing are employed for fair comparison. The best score is marked in bold.*

### TABLE 13
Comparison of Performance on the Test Set of Cityscapes With State-of-the-Art Approaches (Trained on Trainval Set)

| Method                  | Backbone       | Stride | mIoU (%) |
|-------------------------|----------------|--------|----------|
| CCNet [28]              | ResNet-101     | 8×     | 81.90    |
| PSPNet [4]              | ResNet-101     | 8×     | 78.40    |
| PSANet [59]             | ResNet-101     | 8×     | 80.10    |
| OCNet [27]              | ResNet-101     | 8×     | 80.10    |
| OCRNet [26]             | ResNet-101     | 8×     | 81.80    |
| OCRNet [26]             | ResNet-101     | 8×     | 81.50    |
| ACFNet [61]             | ResNet-101     | 8×     | 81.80    |
| ANN [60]                | ResNet-101     | 8×     | 81.30    |
| ACNet [24]              | ResNet-101     | 8×     | 82.30    |
| CPNet [25]              | ResNet-101     | 8×     | 81.30    |
| SPNet [92]              | ResNet-101     | 8×     | 81.80    |
| SPNet [93]              | ResNet-101     | 8×     | 82.00    |
| DenseASPP [58]          | DenseNet-161   | 8×     | 80.60    |
| Dynamic [94]            | Layer33-PSM    | 8×     | 80.70    |
| HRNet [19]              | HRNetV2-W48    | 4×     | 81.60    |
| OCRNet [26]             | HRNetV2-W48    | 4×     | 82.40    |
| DeepLabV3+MCIBI++ ([44]| ResNet-101     | 8×     | 82.03    |
| DeepLabV3+MCIBI++ ([44]| ResNet-St-101  | 8×     | 81.59    |
| DeepLabV3+MCIBI++ ([44]| HRNetV2-W48    | 4×     | 82.55    |
| DeepLabV3+MCIBI++ (ours)| ResNet-101     | 8×     | 82.20    |
| DeepLabV3+MCIBI++ (ours)| ResNet-St-101  | 8×     | 81.70    |
| DeepLabV3+MCIBI++ (ours)| HRNetV2-W48    | 4×     | 82.74    |

*Multi-scale and flipping testing are used for fair comparison. Bold denotes the best score.*
Mult-scale and flipping testing are used for fair comparison.

4.5 Additional Results on Other Benchmarks

Beyond the four challenging image semantic segmentation benchmarks above, we also test MCIBI++ on other benchmarks, including PASCAL-Context [40] and PASCAL VOC 2012 [41], [42].

Results on PASCAL-Context. PASCAL-Context is a challenging scene parsing dataset that provides 4,998 images for training and 5,105 images for validation. We trained the segmentation frameworks for 260 epochs with batch size of 16, crop size of $480 \times 480$ with initial learning rate of 0.004. For evaluation, we performed multi-scale testing with the horizontal flipping operation. As illustrated in Table 14, we can see that UperNet+MCIBI++ with ResNet-101 as the backbone reaches 56.82% mIoU, achieving the new state-of-the-art performance among the ResNet-101-based approaches. This result is also 0.62% mIoU higher than OCRNet with a more robust backbone HRNetV2-W48, which is the previous best segmentor on PASCAL-Context. In addition, we also train UperNet+MCIBI++ with stronger backbone networks, i.e., ResNetSt-101 and Swin-Large and achieve 57.92% and 64.01% mIoU respectively. These results outperform the previous best approach by large margins, i.e., 1.72% mIoU and 7.81% mIoU, respectively. These improvements are nontrivial, given that OCRNet with HRNetV2-W48 is a strong segmentor and the PASCAL-Context task is fairly saturated.

Results on PASCAL VOC 2012. We also evaluate MCIBI++ on the PASCAL VOC 2012 semantic segmentation benchmark, which contains 10,582 (trainaug) training images, 1,449 validation images and 1, 456 testing images. Table 15 shows the comparison results. We report our PyTorch re-implementation results so that all the reported methods can be trained and evaluated on the same environment and settings for a fair comparison. The DeepLabV3 with the ResNet-101 backbone (i.e., 79.52% mIoU) is the previous best algorithm. Our UperNet+MCIBI++ with the same backbone network ResNet-101 obtains a superior mIoU of 80.42% with a significant margin over DeepLabV3, demonstrating the effectiveness of our method.

4.6 Qualitative Results

Visualization. The visual comparisons among FCN, ASPP, ASPP+MCIBI and ASPP+MCIBI++ are shown in Fig. 5. As we can see, ASPP+MCIBI++ achieves better visual results than others. For example, for the first case, only our segmentation framework can extract the pillows from the bed accurately. And for the second case, our algorithm can correctly segment the boat while the other approaches fail to achieve this point.

Visualization of Learned Features. As illustrated in Fig. 6, we visualize the pixel representations learned by the baseline model (left) and our framework (right), respectively. Specifically, we first calculate the composite feature vectors of the pixel representations of the same category within an image. Then, we employ t-SNE algorithm to project these feature vectors into a two-dimensional plane. As we can see, after incorporating the dataset-level information, the features of different categories are more discriminative, i.e., the pixel representations with the same category are more compact and the features of various classes are well separated. This comparison shows that the dataset-level category representations can help augment the pixel representations by improving the spatial distribution of the original features in the non-linear embedding space so that the network can classify the pixel more accurately.
Visualization of Dataset-Level Context Aggregation. Fig. 7 exhibits the visualization results of the proposed dataset-level context aggregation (DCA) scheme on the samples from ADE20K datasets. We can observe that most of the pixels can accurately aggregate the demanded dataset-level category representations, demonstrating the effectiveness of the proposed dataset-level context aggregation scheme.

5 Experiments on Video Semantic Segmentation

In this section, we show the results of our proposed video semantic segmentation paradigm in Section 3.6 (named as Video-MCIBI++ for distinction), where the memory module is also used to integrate the temporal information. Specifically, the proposed Video-MCIBI++ paradigm leverages the memory module to incorporate both the temporal information (i.e., the relations to the previous frames) and the dataset-level contextual information (i.e., the relations to the frames from all trained videos) to the segmentation frameworks.

5.1 Experimental Setup

Dataset. We validate our framework on the recent released VSPW dataset [43]. There are 3,536 annotated videos, including 251,633 frames from 124 categories. The train, validation and test sets of VSPW contain 2,806/343/387 videos with 198,244/24,502/28,887 frames, respectively. It is a quite challenging video semantic segmentation benchmark due to its scale, diversity, long-temporal and high frame rate.

Implementation Details. We trained our segmentors on eight NVIDIA Tesla V100 GPUs with a 32 GB memory per-card. The backbone network was initialized with the weights pretrained on the ImageNet [98] and the newly added modules were randomly initialized. Following [43], all frames in VSPW were resized to keep their shorter side being 512. The optimizer was stochastic gradient descent (SGD) with momentum...
We first show that UperNet+MCIBI++ with ResNet-101 backbone achieves 43.21% mIoU, which outperforms the previous best segmentation framework TCBst-ocr men by 2.84% mIoU, 1.91% TC and 1.51% mVCs, improvements than UperNet+MCIBI with ResNet-101, and UperNet+MCIBI++ with the transformer-based backbone (Swin-Large) have the consistent improvements compared to UperNet+MCIBI+. These results are consistent with the previous observation in Section 4.2, i.e., soft mining contextual information beyond image (MCIBI++) can generate more discriminative dataset-level category representations than mining contextual information beyond image (MCIBI) [45].

### 5.3 Comparison With the State-of-the-Art Methods

Table 17 also compares MCIBI++ and Video-MCIBI++ with the existing image or video semantic segmentation frameworks on the validation set of VSPW. When adopting the ResNet-101 as the backbone, UperNet+MCIBI++ achieves 43.21% mIoU, which outperforms the previous best segmentation framework TCBst-ocr men by 5.39% mIoU. After introducing the temporal information and a more robust backbone network, i.e., Swin-Large, UperNet+Video-MCIBI++ reports the new state-of-the-art results, i.e., 58.88% mIoU, 81.21% TC and 91.53% mVCs.

### 5.4 Quantitative Results

Some segmentation examples from the VSPW validation set are demonstrated in Fig. 8. We can observe that the results from Video-MCIBI++ (i.e., the third row) are more stable and accurate than MCIBI++ (i.e., the second row). For example, in the first video (i.e., the first row), MCIBI++ labels one part of the bed as the other categories while Video-MCIBI++ can predict the bed accurately and stably.

### 6 Application Discussion

In this section, we discuss some applications of our proposed MCIBI++ paradigm and some potential issues we should pay attention to. The first point is the potential domain divergence problem of different datasets. For instance, if the trained dataset only contains the single-person scenes, the dataset-level
distribution information of each category in the memory module may be insufficient for the multi-person scenes. Although the results in Table 8 show that the memory module can carry some high-level information to benefit the human parsing in different domains, the overall segmentation performance of the segmentors trained on the single-person scenes is still far lower than the segmentors which are trained on the multi-person scenes directly. Therefore, it is still necessary to design other strategies to tackle the domain divergence. For example, during the network training, we suggest that the users can conduct some data augmentation [99] to generate the training images of various domains so that the memory module can better adopt to the realistic scene. Moreover, we find that some recent works [100] have followed our previous study (i.e., the conference version) and designed some strategies to store the domain-agnostic information into the memory to achieve domain adaptation, which may be also very helpful for the application of our method. The second point we should pay attention to is that introducing memory module may make the network vulnerable from the adversarial attacks. To alleviate the problem, we suggest that the users can also apply some data augmentation strategies [101] during training to enhance the generation ability of the memory module. Overall, we expect that proposed MCIBI++ can shine in the real applications.

7 Conclusion and Outlook

This paper presents a new perspective for tackling the context aggregation in semantic segmentation. We propose to
mine the contextual information beyond the input image in a soft way (MCIBI++) to further improve the pixel representations. A memory module is set to store the dataset-level distribution information of various categories and a dataset-level context aggregation scheme is designed to aggregate the dataset-level category representations yielded by the stored distribution information for each pixel according to their class probability distribution. At last, the aggregated dataset-level contextual information is leveraged to augment the original pixel representations for more accurate prediction. In addition, a coarse-to-fine iterative inference strategy is proposed to further boost the segmentation performance by progressively improving the accuracy of aggregate weights. Also, MCIBI++ is generalized to the video semantic segmentation task by incorporating the temporal information in the process of gathering the dataset-level category representations for the pixels. Extensive experiments demonstrate the effectiveness of our method.

Comparing Tables 1 and 3, we can observe that the achieved segmentation performance is far below the theoretical performance limit of our framework (43.74% versus 68.82%). We thus believe that our future works can focus on how to aggregate more accurate dataset-level category representations for the pixels, i.e., how to improve the dataset-level context aggregation scheme. In addition, modeling the dataset-level category representations better and updating the memory module more efficiently will also be further investigated in our future work.

REFERENCES

[1] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. Int. Conf. Learn. Representations, 2015.
[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–778.
[3] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, “GCNet: Non-local networks meet squeeze-excitation networks and beyond,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops, 2019, pp. 1971–1980.
[4] F. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 2881–2890.
[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 4, pp. 834–848, Apr. 2018.
[6] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” 2017, arXiv:1706.05587.
[7] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 801–818.
[8] J. Fu et al., “Dual attention network for scene segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 3146–3154.
[9] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 7794–7803.
[10] M. Bertinetto, A. R. Tikhi, and M. B. Blaschko, “The lovasz-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 4413–4421.
[11] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 3431–3440.
[12] H. K. Cheng, J. Chung, Y.-W. Tai, and C.-K. Tang, “CascadeSP: Toward class-agnostic and very high-resolution segmentation via global and local refinement,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 8890–8899.
[13] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Semantic image segmentation with deep convolutional nets and fully connected CRFs,” in Proc. Int. Conf. Learn. Representations, 2015.
[14] S. Zheng et al., “Conditional random fields as recurrent neural networks,” in Proc. IEEE Int. Conf. Comput. Vis., 2015, pp. 1529–1537.
[15] A. Kirillov, Y. Wu, K. He, and R. Girshick, “PointRend: Image segmentation as rendering,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 5979–5988.
[16] T. Yuan, T. Liu, Z. Wei, X. Wei, S. Wei, and Y. Zhao, “Devil in the details: Towards accurate single and multiple human parsing,” in Proc. AAAI Conf. Artif. Intell., 2019, pp. 4814–4821.
[17] F. Yu, V. Koltun, and T. Funkhouser, “Dilated residual networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 472–480.
[18] H. Zhang et al., “ResNeSt: Split-attention networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 2736–2746.
[19] H. Zhang et al., “Deep high-resolution representation learning for visual recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 10, pp. 3349–3364, Oct. 2021.
[20] Z. Liu et al., “Swin transformer: Hierarchical vision transformer using shifted windows,” in Proc. IEEE Int. Conf. Comput. Vis., 2021, pp. 10012–10022.
[21] S. Zheng et al., “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 6681–6689.
[22] B. Cheng, A. Schwing, and A. Kirillov, “Per-pixel classification is not all you need for semantic segmentation,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 17 864–17 873.
[23] H. K. Cheng, J. Chung, Y.-W. Tai, and C.-K. Tang, “CascadePSP: Toward class-agnostic and very high-resolution segmentation via global and local refinement,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 7794–7803.
[24] S. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance normalization: Improving the performance of deep neural networks,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 418–426.
[25] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep neural networks,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 448–456.
[26] S. Zheng et al., “Conditional random fields as recurrent neural networks,” in Proc. IEEE Int. Conf. Comput. Vis., 2015, pp. 448–456.
[27] A. Kirillov, Y. Wu, K. He, and R. Girshick, “PointRend: Image segmentation as rendering,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 5979–5988.
[28] S. Zheng et al., “Conditional random fields as recurrent neural networks,” in Proc. IEEE Int. Conf. Comput. Vis., 2015, pp. 448–456.
[29] Y. Wu and K. He, “Group normalization,” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 1938–1946.
[30] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep neural networks,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 448–456.
[31] Y. Wu and K. He, “Group normalization,” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 1938–1946.
[32] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, “CCNet: Criss-cross attention for semantic segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 7189–7198.
[33] Z. Jin, B. Liu, Q. Chu, and N. Yu, “ISNet: Integrate image-level and semantic-level context for semantic segmentation,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 7189–7198.
[34] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, “GCNet: Non-local networks meet squeeze-excitation networks and beyond,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops, 2019, pp. 1971–1980.
[35] T.-W. Ke, J. Li, B. Liu, Z. Liu, and S. X. Yu, “Adaptive affinity fields for semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 587–602.
[36] S. Zhao, Y. Wang, Z. Yang, and D. Cai, “Region mutual information loss for semantic segmentation,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 11 117–11 127.
[37] J. Li, B. J. K. Rikos, and G. E. Hinton, “Layer normalization,” 2016, arXiv:1607.06450.
[38] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 448–456.
[39] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, “OCNet: Object context for semantic segmentation,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 11 117–11 127.
[40] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, “OCNet: Object context for semantic segmentation,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 11 117–11 127.
[41] J. Li, B. J. K. Rikos, and G. E. Hinton, “Layer normalization,” 2016, arXiv:1607.06450.
Z. Zhang, C. Su, L. Zheng, X. Xie, and Y. Li, “On the correlation among edge, pose and parsing,” *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Sep. 01, 2021, doi: 10.1109/TPAMI.2021.3108771.

H. Ding, X. Jiang, B. Shuai, A. Q. Liu, and G. Wang, “Semantic correlation promoted shape-variant context for segmentation,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8883–8894.

X. Li et al., “Semantic flow for fast and accurate scene parsing,” in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 775–793.

Q. Hou, L. Zhang, M.-M. Cheng, and J. Feng, “Strip pooling: Rethinking spatial pooling for scene parsing,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4003–4012.

Y. Li et al., “Learning dynamic routing for semantic segmentation,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8553–8562.

A. Kirillov, R. Girshick, K. He, and P. Dollar, “Panoptic feature pyramid networks,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6392–6401. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2019.00656

L. Huang, Y. Yuan, J. Guo, C. Zhang, X. Chen, and J. Wang, “Interlaced sparse self-attention for semantic segmentation,” 2019, arXiv:1907.12273.

M. Yin et al., “Disentangled non-local neural networks,” in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 191–207.

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.

Y. Zou, Z. Zhang, C.-L. Li, H. Zhang, T. Pfister, and J.-B. Huang, “Learning instance-specific adaptation for cross-domain segmentation,” 2022, arXiv:2203.16530.

J. Kim, J. Lee, J. Park, D. Min, and K. Sohn, “Pin the memory: Learning to generalize semantic segmentation,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 4350–4360.

V. Zantedeschi, M.-I. Nicolae, and A. Kawat, “Efficient defenses against adversarial attacks,” in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, 2017, pp. 39–49.

Zhenchao Jin received the MS degree from the CAS Key Laboratory of Electromagnetic Space Information, Electronic Engineering and Information Science, University of Science and Technology of China, Heifei, China. He is currently working toward the PhD degree with the Department of Statistics and Actuarial Science, The University of Hong Kong. His research interests focus on semantic segmentation and object detection.

Dongdong Yu received the PhD degree in pattern recognition and intelligent systems from the Key Laboratory of Molecular Imaging, Institute of Automation, Chinese Academy of Sciences, Beijing, China. He is a researcher with OPPO research. His research interests focus on image classification, human keypoint, scene parsing and object detection.

Zehuan Yuan received the BS and PhD degrees both from the Department of Computer Science and Technology, Nanjing University, China, in 2015, and the PhD degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, in 2019. He conducted his postdoctoral training with Stanford University during 2019-2021. He is currently a researcher in Bytedance AI Lab. He has published more than fifteen academic papers on the top-tier international journals and conferences, such as ICML, CVPR, IJCAI, AAAI, ICLR, ECCV, etc. His research interests lie in computer vision and machine learning.

Lequan Yu (Member, IEEE) received the BEng degree from the Department of Computer Science and Technology, Zhejiang University, Hangzhou, China, in 2015, and the PhD degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, in 2019. He conducted his postdoctoral training with Stanford University during 2019-2021. He is currently an assistant professor with the Department of Statistics and Actuarial Science, The University of Hong Kong. His research lies at medical image analysis, computer vision, machine learning and AI in healthcare.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.