SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2024, Vol. 67, Iss. 9, 192105:1–192105:20
https://doi.org/10.1007/s11432-023-4061-2

Relative difficulty distillation for semantic segmentation

Dong LIANG[1,2], Yue SUN[1], Yun DU[1], Songcan CHEN[1*] & Sheng-Jun HUANG[1]

[1]*MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing* 211106*, China;*
[2]*Shenzhen Research Institute, Nanjing University of Aeronautics and Astronautics, Shenzhen* 518000*, China*

Received 5 September 2023/Revised 2 January 2024/Accepted 16 January 2024/Published online 20 August 2024

Abstract Current knowledge distillation (KD) methods primarily focus on transferring various structured knowledge and designing corresponding optimization goals to encourage the student network to imitate the output of the teacher network. However, introducing too many additional optimization objectives may lead to unstable training, such as gradient conflicts. Moreover, these methods ignored the guidelines of relative learning difficulty between the teacher and student networks. Inspired by human cognitive science, in this paper, we redefine knowledge from a new perspective — the student and teacher networks' relative difficulty of samples, and propose a pixel-level KD paradigm for semantic segmentation named relative difficulty distillation (RDD). We propose a two-stage RDD framework: teacher-full evaluated RDD (TFE-RDD) and teacher-student evaluated RDD (TSE-RDD). RDD allows the teacher network to provide effective guidance on learning focus without additional optimization goals, thus avoiding adjusting learning weights for multiple losses. Extensive experimental evaluations using a general distillation loss function on popular datasets such as Cityscapes, CamVid, Pascal VOC, and ADE20k demonstrate the effectiveness of RDD against state-of-the-art KD methods. Additionally, our research showcases that RDD can integrate with existing KD methods to improve their upper performance bound. Codes are available at https://github.com/sunyueue/RDD.git.

Keywords knowledge distillation, semantic segmentation, relative difficulty, sample weighting, prediction discrepancy

1 Introduction

Semantic segmentation is the basis of many vision-understanding systems, e.g., medically assisted diagnosis [1–5], embodied AI [6–10], and driving assistant [11–15]. Current deep neural networks, e.g., DeepLab series [16–19], PSPNet [20], HRNet [21], have achieved remarkable success. However, these cumbersome models often lead to expensive computation costs. To address this issue for embedded and edge deployments, researchers have focused on designing lightweight networks, e.g., ENet [22], ICNet [23], BiSeNet [24] and ESPNet [25]. These networks employ techniques like model quantization [26] and pruning [27] to reduce inference cost or utilize knowledge distillation (KD) [28,29] to transfer the capabilities of larger models to lightweight ones.

Current KD methods [28–31] primarily focus on the design of feature/response-based knowledge and optimization objectives (i.e., distillation loss), aiming to make the student network mimic the teacher's output. However, additional optimization objectives accompanied by an imbalance problem may intuitively lead to training instability [32, 33]. Furthermore, these approaches often neglect the teacher's ability to delineate learning focus and guide the student network's learning according to the samples' learning difficulty.

In the field of human education and cognition, teachers assess the difficulty of learning materials based on students' needs and adjust the learning difficulty according to their mastery level [34]. This personalized instruction enables students to face appropriate challenges and facilitates the development

* Corresponding author (email: s.chen@nuaa.edu.cn)

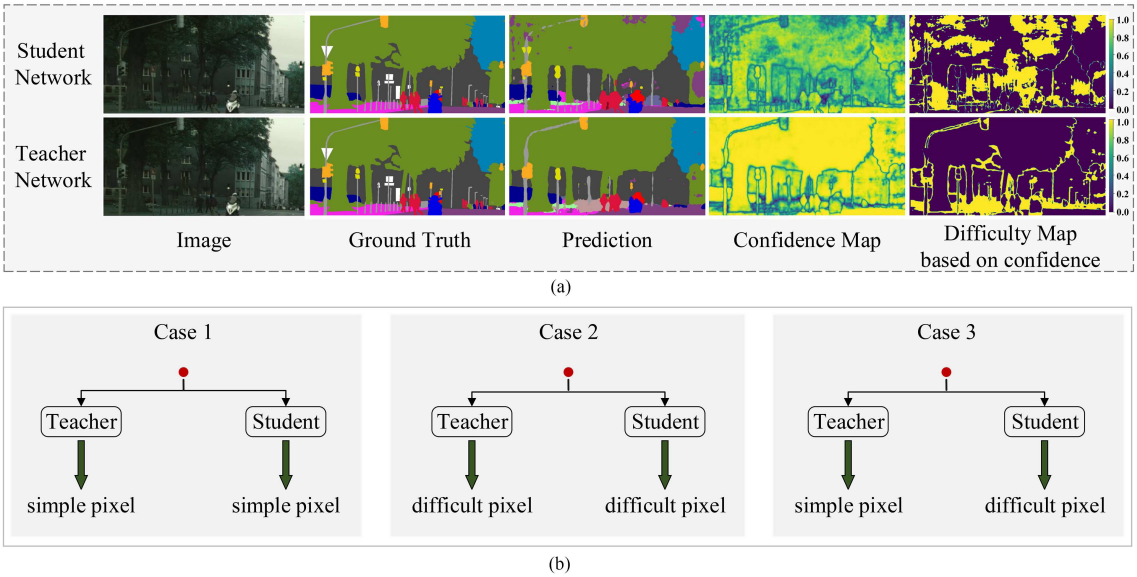

Figure 1 Prediction discrepancy based on confidence between the teacher and student networks is used to assess the learning difficulty of pixels. (a) The formation process of difficulty maps based on confidence for the student and teacher networks. (b) There are three situations in which the student and teacher networks evaluate the difficulty of a pixel: Case 1, where both networks evaluate the pixel as easy; Case 2, where both networks evaluate the pixel as difficult; Case 3, where there is disagreement on the difficulty of the pixel. Note that for Case 3, the pixel with prediction discrepancies is the difficult pixel that the student network should learn.

of higher-order cognitive processes. From this perspective, we argue that the guidance provided by the teacher network in sample selection and sample difficulty adjustment is equally important for training the student network. By incorporating this guidance into KD, the accuracy and robustness of student networks can be improved.

Some previous studies [35–37] have evaluated sample difficulty using confidence measures/loss and selectively trained samples based on their difficulty levels. They consider high-loss/low-confidence samples difficult and low-loss/high-confidence samples easier. As depicted in Figure 1(a), for each pixel in an image, the network utilizes the maximum probability of predicted classes as a confidence value to filter difficult pixels, resulting in a confidence map representing the pixel-wise confidence scores. Subsequently, using a predefined threshold, pixels with confidence values lower than the threshold are marked as 1 (indicating difficult pixels), and those with values higher than the threshold are marked as 0 (indicating simple pixels), thereby generating a difficulty map based on confidence. The network's performance and generalization capabilities can be enhanced by utilizing this difficulty map to select challenging pixels for training. However, adding too many difficult pixels can also degrade network performance. Therefore, determining which difficult pixels should be mined becomes an essential topic of discussion.

In Figure 1(b), we show three situations in which teacher and student networks evaluate the difficulty of a pixel. The pixel is evaluated to be easy by both the teacher and student network (as shown in Case 1), which is usually the pixel the student network has mastered and no longer needs to spend time on learning. On the other hand, due to the strait of the semantic segmentation fine-annotation process, the difficult pixel that the teacher network cannot master may even be noisy annotations or have semantic ambiguities. The student network may also struggle to accurately predict the challenging pixel (as shown in Case 2). This class of difficult pixels should also not be learned. An ideal valuable difficulty pixel is one that the student network considers difficult and has not fully mastered but is expected to learn well (as shown in Case 3), and the prediction discrepancy based on confidence between the teacher and student networks is used to generate the relative difficulty of pixels can provide such information. Therefore, we define knowledge from a new perspective — the teacher network poses the sample's pixel-level relative learning difficulty as knowledge, providing a guide of divergent samples mining for the learning of the student network. We also avoid constructing additional optimization objectives to ease instability in network training due to multiple optimization objectives. It also allows integration with other feature/response-based KD methods if desired.

We further discovered that relative difficulty should be in different forms in the learning life cycle.

In the early stage of human education [38, 39], teachers employ direct instruction to provide students with explicit guidance and a structured learning process, enabling them to quickly acquire fundamental knowledge and skills. Similarly, in the early learning stage of KD, relying on the student network's judgments to guide training may transmit and amplify errors, making it challenging to correct the student network. In contrast, the supervisory information from the teacher network is more beneficial for the student network to learn relatively simple pixels and achieve rapid convergence. On the other hand, adaptive teaching [40, 41] and personalized teaching [42] in human education emphasize that teachers should customize the learning process according to learners' current levels and needs to adapt to individual differences and learners' abilities. Similarly, in KD, the student network gains judgment abilities as it progresses through the training stages. The student network can assess the sample's difficulty together with the teacher network and adjust its learning accordingly. This collaborative approach provides a comprehensive perspective and accurate assessment of sample difficulty, thereby effectively guiding the network to learn relatively difficult pixels in the later learning stage to improve the upper performance bound.

Based on the above discussion, we propose a pixel-level KD paradigm named relative difficulty distillation (RDD) for semantic segmentation. RDD includes two specific distillation methods: teacher-full evaluated RDD (TFE-RDD) and teacher-student evaluated RDD (TSE-RDD), which are two successive learning stages. TFE-RDD is designed for the early learning stage when the student network is still far from converging. It leverages prediction discrepancy between the primary and auxiliary classifiers in the teacher network to acquire relative difficulty knowledge. This knowledge guides the student network to focus on pixels with lower difficulty. As the learning progresses, TSE-RDD comes into play during the later learning stage. It leverages the prediction discrepancy between the student and teacher networks to generate reliable relative difficulty knowledge. This knowledge progressively guides the student network toward mastering difficult pixels over time. The proposed RDD scheme effectively distills pixel-level relative difficulty as dark knowledge, guiding the student network to focus more on informative and representative pixels at appropriate learning stages.

The main contributions of this work can be summarized as follows:

• We propose a new KD paradigm for semantic segmentation named RDD. RDD avoids additional optimization objectives and can seamlessly integrate with other feature/response-based KD methods to improve their upper performance bound.

• We devise two specific RDD methods, TFE-RDD and TSE-RDD, tailored for the early and later learning stages, respectively. The teacher network incorporates the student network to generate relative difficulty, guiding the student network to focus on the most valuable pixels during the different learning stages.

• RDD achieves the best distillation performance among the state-of-the-art methods on four popular semantic segmentation datasets with various semantic segmentation architectures.

2 Related work

2.1 KD for semantic segmentation

KD [30] is a method that utilizes the output's probability distribution of the teacher network as the training target of the student network. Methods with similar ideas can be traced back to [43], which first train an ensemble model and then utilize the reliable output generated by the ensemble model to replace the labels of the original training samples so that the target network can learn. In order to deploy deep models on devices with limited resources, Buciluǎ et al. [44] first proposed model compression technology, which can transfer information from large models or ensemble models to small models for training without significantly reducing accuracy. Subsequently, Hinton et al. [30] summarized and developed this idea and formally proposed the concept of KD.

Previous KD algorithms [30, 45–53] primarily focused on image-level classification tasks. However, image-level KD seldom considers the locally structured information, so it is congenitally deficient for pixel-level semantic segmentation. To alleviate this problem, Liu et al. [28] proposed two structured distillation schemes to transfer structured knowledge from the teacher network to the student network. Wang et al. [54] introduced intra-class feature variation distillation (IFVD) to transform the teacher network into the student network. Shu et al. [31] utilized channel-wise KD to minimize the disparity in

normalized channel activations between the teacher and the student networks. Zheng et al. [55] proposed an uncertainty-aware pseudo-label learning approach that leverages uncertainty estimation and KD to rectify noisy pseudo labels for domain adaptive semantic segmentation. Holder et al. [56] presented an efficient uncertainty estimation method for semantic segmentation by transferring uncertainty knowledge from teacher to student networks via distillation. Ji et al. [57] proposed a contourlet decomposition module (CDM) and a denoised texture intensity equalization module (DTIEM) to mine the structural texture knowledge and enhance the statistical texture knowledge, respectively. To establish the global semantic relationships between pixels across different images, Yang et al. [29] attempted to leverage pixel-to-pixel and pixel-to-region relationships as knowledge and transfer global pixel correlation from teachers to students.

The above KD methods mainly transfer various feature/response-based structured knowledge by introducing additional optimization objectives. For example, in [57], the author provided five loss terms to obtain knowledge from low-level and high-level features and force the student network to imitate the teacher network from a broader perspective. Similarly, CIRKD [29] also provided five loss terms to learn more comprehensive cross-image pixel dependencies. However, in this multi-objective optimization problem, there are mutual constraints between various objectives, making it difficult for the model to accurately fit the training data, leading to issues such as training instability [32, 33]. Moreover, more computation and storage time may also be required when computing the structured knowledge of the teacher network, thus slowing down the distillation process. For example, CIRKD [29] utilized a memory bank for comparative learning, which consumes a certain amount of time when storing pixel embeddings. In contrast, RDD achieves knowledge transfer by scaling relative gradients without introducing additional optimization objectives, avoiding the trade-off problem between multiple objectives.

2.2 Training strategies based on curriculum learning

Curriculum learning (CL) [58] is a training strategy that trains a machine learning model from simple samples to difficult samples by mimicking the meaningful learning sequence in a human curriculum. CL can be extended to other methods. For example, self-paced learning [59] refers explicitly to a training strategy in which the student network acts as a teacher network and measures the difficulty of training samples based on its losses. The teacher network is pre-trained in the current dataset or other large-scale datasets, and its knowledge is transferred to the curriculum design of the student network. Compared with self-paced learning [59], transfer teacher [60] selects a mature teacher network to evaluate the difficulty of the training samples. This method addresses the limitation that "the machine learning network itself may not be mature enough in the initial stage of training to measure sample difficulty accurately". Although these CL methods have shown effectiveness and ease of use, they often come with increased training costs. Specifically for semantic segmentation tasks, designing courses or curricula with samples sorted from easy to difficult can lead to additional computational overhead. Additionally, it has been observed in previous studies [61] that CL may lead to the loss of correlation between some samples when applied alongside KD methods based on sample relationships or graphs. This correlation loss can result in model degradation.

2.3 Training strategies based on adaptive sample reweighting

The concept of reweighting each sample has been extensively studied in the literature. Some studies [62,63] claimed that encouraging models to learn from difficult samples can enhance their upper performance bound. For example, Freund et al. [63] selected more difficult samples to train subsequent classifiers. OHEM [35] utilizes the most difficult samples, with the idea that high-loss samples can better train the classifier. Some studies [62, 64, 65] proposed soft sampling methods that train the network based on sample importance. Focal loss [62] dynamically assigns higher weights to difficult samples. Prime sample attention [64] analyzes evaluation indicators of object detection and assigns weight to positive and negative samples according to different criteria. However, Li et al. [65] argued that outliers also exist among difficult instances and introduced the metric of gradient density to adjust the data distribution. Since the above methods primarily focus on addressing class imbalance problems, they prioritize difficult samples with large training losses and overlook the contribution of simple samples.

In this paper, we similarly incorporate the idea of sample reweighting to generate reliable relative difficulty knowledge by exploiting prediction disagreement between student and teacher networks. Weighting

the samples based on generated relative difficulty knowledge can guide the student network to focus its learning on valuable easy or difficult pixels at different stages of training.

3 Methodology

In this section, we show RDD-based semantic segmentation through the relative difficulty knowledge RD provided by the teacher and student networks. We begin by introducing the loss function paradigm for the semantic segmentation with KD in Subsection 3.1. Next, we present the generalized definition of the relative difficulty knowledge RD in Subsection 3.2. Subsequently, we defined the two-stage distillation based on relative difficulty and the corresponding generated knowledge $\mathrm{RD}_{\mathrm{TFE}}$ and $\mathrm{RD}_{\mathrm{TSE}}$ in Subsection 3.3, respectively. The implementation of TFE-RDD and TSE-RDD are in Subsections 3.4 and 3.5.

3.1 Preliminary

Loss function paradigm for semantic segmentation. Semantic segmentation classifies each pixel in an image from C categories to corresponding category labels. The segmentation network typically takes an RGB image with dimensions $W \times H \times 3$ as input. It extracts the dense image feature map F using a backbone architecture, where H and W represent the height and width of the image. The categorical logit map $\boldsymbol{Z}$ is generated from feature map F using the classifier by applying a classifier. Optimization is then performed using a cross-entropy loss:

$$L_{\mathrm{task}} = \frac{1}{H \times W} \sum_{h=1}^{H} \sum_{w=1}^{W} \mathrm{CE}\left(\sigma\left(\boldsymbol{Z}_{h,w}\right), \boldsymbol{y}_{h,w}\right), \tag{1}$$

where $\boldsymbol{y}_{h,w}$ denotes the ground-truth label for the (h,w)-th pixel, and $\boldsymbol{Z}_{h,w}$ denotes the output logits for the (h,w)-th pixel. The softmax function σ generates the category probability, and CE is the cross-entropy loss to measure the difference between the ground truth and category probability.

Loss function paradigm for pixel-wise KD. KD in semantic segmentation tasks generally employs a point-wise alignment to learn structural knowledge among spatial locations. It can be formulated as follows:

$$L_{\mathrm{kd}} = \frac{1}{H \times W} \sum_{h=1}^{H} \sum_{w=1}^{W} \mathrm{KL}\left[\sigma\left(\frac{\boldsymbol{Z}^{s}_{h,w}}{T}\right) \middle\| \sigma\left(\frac{\boldsymbol{Z}^{t}_{h,w}}{T}\right)\right], \tag{2}$$

where $\boldsymbol{Z}^{s}_{h,w}$ and $\boldsymbol{Z}^{t}_{h,w}$ represent the output logits for the (h,w)-th pixel produced from the student and the teacher network, respectively. σ function calculates the category probability of the (h,w)-th pixel generated by the student and teacher networks, respectively. KL denotes the Kullback-Leibler divergence, which measures the difference between two probability distributions. The parameter T represents the temperature taken by distillation and reflects the label's softening degree. For a fair comparison with previous studies [28, 29], we set T =1 in our experiments.

3.2 Define relative difficulty as knowledge

A well-trained teacher network can provide the student network with relative difficulty knowledge, denoted as $\mathrm{RD} \in \mathbb{R}^{H \times W}$, where $\mathbb{R}$ represents the euclidean space. We use RD to measure the difficulty of samples. The value of RD indicates the level of difficulty associated with each sample. Larger values correspond to more difficult samples. We apply the proposed RD to the task loss L_{task} calculation. The loss L_{task} with relative difficulty knowledge RD can be formalized as $w(L_{\mathrm{task}}, \mathrm{RD})$. w imposes sample weights on L_{task} based on the RD. Intuitively, RD increases the loss contribution related to the valuable pixel sample in the current stage without introducing additional training objectives. The final KD loss is formulated as

$$L_{\mathrm{RDD}} = w(L_{\mathrm{task}}, \mathrm{RD}) + L_{\mathrm{kd}}, \tag{3}$$

where w leverages RD to guide learning during the calculation of L_{task}, and L_{kd} provides additional supervision. The w function and L_{kd} loss represent concrete implementations of difficulty-based distillation and feature/response-based distillation, respectively. These two distillation schemes are independent and can be integrated additively. In our proposed method, the cross-entropy loss is used for calculating L_{task}, and L_{kd} refers to the distillation loss mentioned in Subsection 3.1.

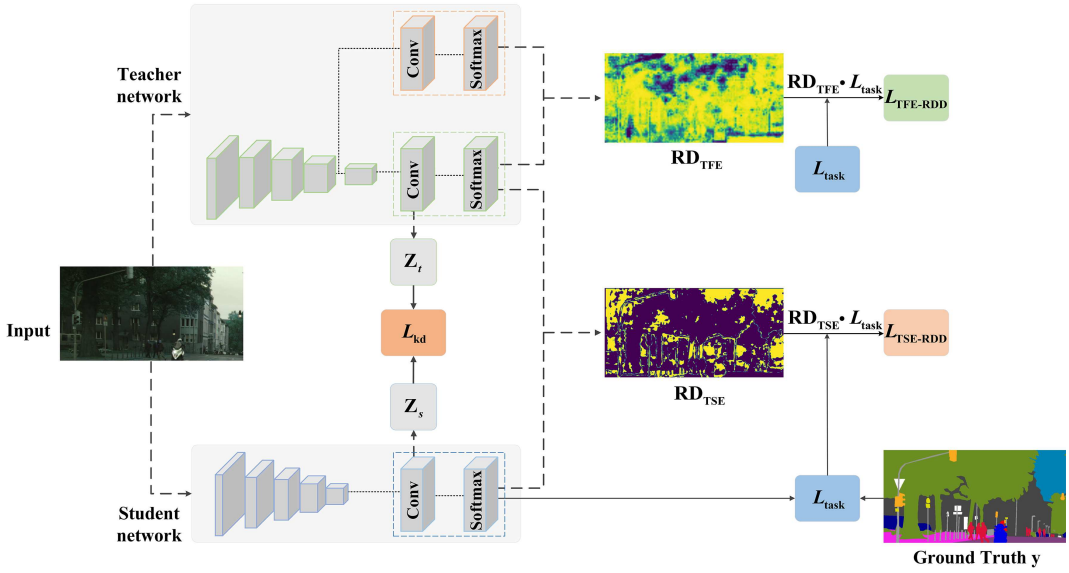

Figure 2 Proposed RDD.

3.3 Define two-stage RDD

During the initial stages of training, the student network lacks the ability to accurately predict pixels and requires full guidance from the teacher network to determine the learning difficulty. The student network achieves fast convergence by focusing on relatively simple pixels. As dynamic learning progresses, the student network gradually becomes capable of accurately fitting most simple pixels and develops its own judgment capabilities. At this stage, the teacher network needs to select appropriate, challenging pixels based on the student network's needs and skill level to promote the development of the student network's capabilities. Consequently, we believe RD should be diversified and progressive. Based on the above inspiration, we propose two stages of RDD: TFE-RDD in the early learning stage and TSE-RDD in the later learning stage. TFE-RDD and TSE-RDD provide two types of relative difficulty knowledge: teacher-full evaluated relative difficulty (denoted as $\mathrm{RD}_{\mathrm{TFE}}$) and teacher-student evaluated relative difficulty (denoted as $\mathrm{RD}_{\mathrm{TSE}}$) respectively.

As depicted in Figure 2, in the TFE-RDD stage, RDD relies on the teacher network to provide full guidance on the learning difficulty. The prediction discrepancy between the teacher network's primary and auxiliary classifiers can provide $\mathrm{RD}_{\mathrm{TFE}}$ to the student network, which uses $\mathrm{RD}_{\mathrm{TFE}}$ to focus training on simple pixels. We incorporate $\mathrm{RD}_{\mathrm{TFE}}$ into L_{task} calculation as follows:

$$L_{\text{TFE-RDD}} = \mathrm{RD}_{\mathrm{TFE}} \cdot L_{\mathrm{task}}. \tag{4}$$

As the student network matures, it becomes necessary for the teacher network to consider the student's relative difficulty in teaching according to their disagreement. In the TSE-RDD stage, we believe that leveraging the discrepancy between the teacher and student networks' prediction confidences will generate $\mathrm{RD}_{\mathrm{TSE}}$. The student network utilizes $\mathrm{RD}_{\mathrm{TSE}}$ to focus training on valuable difficult pixels. We apply $\mathrm{RD}_{\mathrm{TSE}}$ to impose learning attention during L_{task} calculation as follows:

$$L_{\text{TSE-RDD}} = \mathrm{RD}_{\mathrm{TSE}} \cdot L_{\mathrm{task}}. \tag{5}$$

The following sections provide further details about TFE-RDD and TSE-RDD separately.

3.4 TFE-RDD

Most semantic segmentation networks incorporate double classification heads (primary and auxiliary classifier) to alleviate the gradient disappearance issue, including [16, 17, 20] and the modified DeepLab-V2 in [66–68]. Previous studies [69, 70] have shown that a difference in predictions between the primary and auxiliary classifiers indicates the sample is relatively difficult to classify.

In the left part of Figure 3, we present a simplified schematic diagram of a dual classifier network based on DeepLabV3 [17] with ResNet-101 [71] as its backbone. The ResNet-101 network consists of

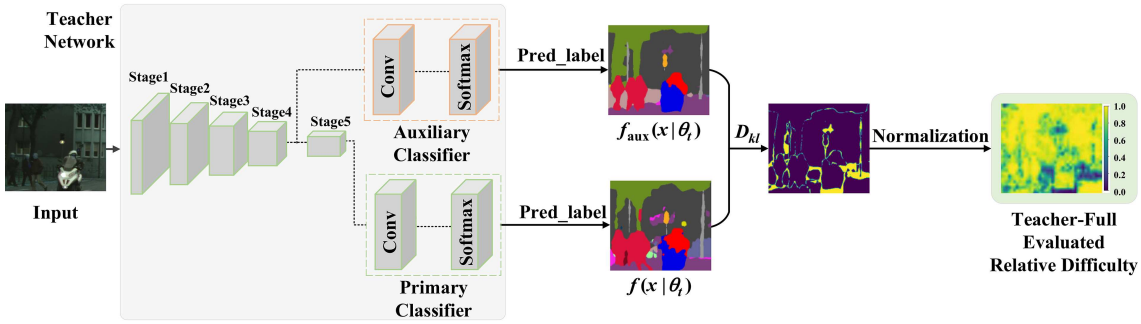

Figure 3 In the TFE-RDD stage, the difficulty map based on prediction discrepancy is obtained using the prediction results of primary and auxiliary classifiers of the teacher network, and the student network is guided to learn simple pixels for efficient fitting.

five stage blocks, which produce five feature maps capturing different scales and semantic information. We introduce an auxiliary classifier with an identical structure to the primary classifier, comprising a convolutional kernel and softmax function. The primary classifier employs the feature map acquired from stage 5 as input, while the auxiliary classifier uses the feature map obtained from stage 4. Since each stage's feature map corresponds to distinct semantic information and context, both classifiers classify the input based on features from different stages. Any differences in these features will result in different predictions. Difficult pixels often exhibit more significant differences in feature maps across different stages, as they have more complex feature representations. Consequently, the two classifiers may produce inconsistent predictions for these challenging pixels.

In the early stage of training, the student network has not yet converged, and the prediction discrepancy between the student network's double classifiers is not a criterion for determining whether the sample is difficult to classify. Therefore, the pre-trained teacher network is more suitable for guiding the student network in the early learning stage. We leverage the prediction discrepancy between the teacher network's primary and auxiliary classifiers to reflect the difficulty of pixels.

We first utilize the teacher network's primary classifier $f(\cdot \mid \theta_t)$ and auxiliary classifier $f_{\text{aux}}(\cdot \mid \theta_t)$ to predict the label of each pixel. The KL divergence (D_{kl}) is then utilized to calculate the discrepancy between the two predicted labels:

$$D_{\text{kl}} = f(x \mid \theta_t) \log \left(\frac{f(x \mid \theta_t)}{f_{\text{aux}}(x \mid \theta_t)} \right), \tag{6}$$

where θ_t represents the parameter set of teacher network. After obtaining the D_{kl} for each pixel via KL divergence calculation, we normalize it within a $[0, 1]$ range to obtain the pixel's relative difficulty values. These values are referred to as teacher-full evaluated relative difficulty (RD_{TFE}), which is calculated as follows:

$$\text{RD}_{\text{TFE}} = \exp\{-D_{\text{kl}}\}. \tag{7}$$

The value of RD_{TFE} reflects how difficult each pixel is classified, and larger values indicate simpler pixels. Subsequently, we apply RD_{TFE} to the semantic segmentation task loss (L_{task}) to get the total loss for the current stage using the following formula:

$$L_{\text{RDD}} = L_{\text{TFE-RDD}} + L_{\text{kd}} = \text{RD}_{\text{TFE}} \cdot L_{\text{task}} + L_{\text{kd}}. \tag{8}$$

By adjusting the loss weight for each pixel based on RD_{TFE}, simple pixels have larger values of RD_{TFE} for the student network to learn in this stage.

3.5 TSE-RDD

As the student network gradually converges, difficult pixels become more helpful in further improving the network's performance. However, it is essential to note that adding too many difficult pixels can adversely affect the student network's performance. According to observations from previous studies [72, 73], deep learning networks often suffer from overconfidence and tend to overfit difficult pixels. Additionally, after the initial training stage, the student network develops an ability to judge the difficulty of samples to some extent. Relying solely on the relative difficulty provided by the teacher network does not consider the student network's needs. Hence, it is crucial for the teacher network to assess the sample's learning

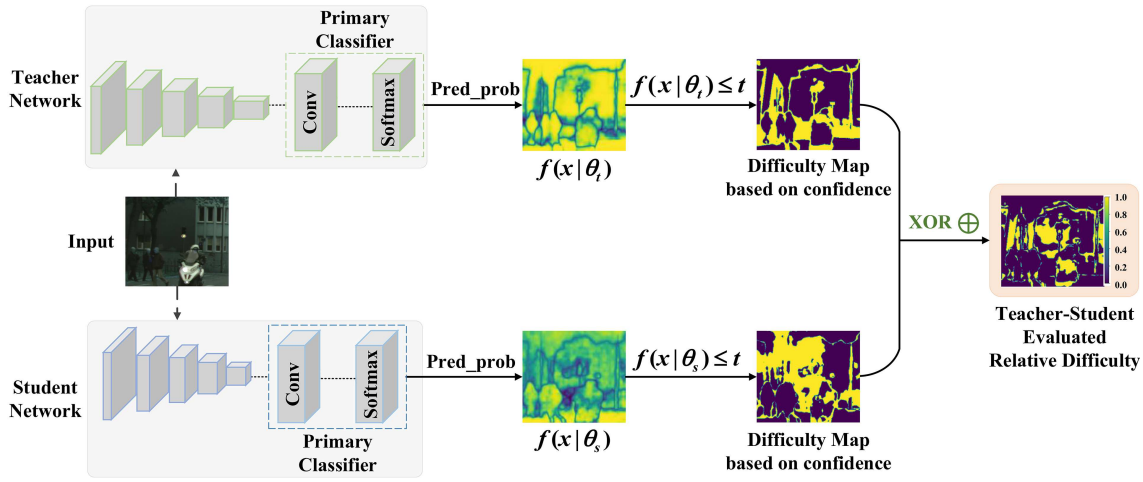

Figure 4 In the TSE-RDD stage, the difficulty maps based on confidence are obtained using the prediction results of the teacher and student networks. The filtered difficulty maps are applied to Exclusive-OR operations to obtain valuable difficult pixels and expand the upper performance bound.

difficulty based on the student network's requirements and adjust the learning difficulty according to their disagreements.

Considering the aforementioned factors, we propose a method that utilizes the relative difficulty generated by the discrepancy based on prediction confidence between the teacher and student networks, as depicted in Figure 4. First, the teacher and student networks discard pixels with confidence values greater than a threshold t while retaining those difficult pixels that still need to be learned. This filtering process generates the corresponding difficulty maps based on confidence. We then perform an XOR operation between the filtered difficulty maps of both networks to obtain a relative difficulty map named $\mathrm{RD}_{\mathrm{TSE}}$. The formulation for computing $\mathrm{RD}_{\mathrm{TSE}}$ is as follows:

$$\mathrm{RD}_{\mathrm{TSE}} = (f(x \mid \theta_s) \leqslant t) \oplus (f(x \mid \theta_t) \leqslant t). \tag{9}$$

$\mathrm{RD}_{\mathrm{TSE}}$ takes a value of 0 or 1. A value of 0 indicates two cases. (1) The current pixel is simple and can be easily mastered by both the teacher and student networks. Such pixels can be unlearned at the current stage. (2) The current pixel is difficult for both networks to master. Difficult pixels that even the teacher network cannot master may contain noisy annotations or semantic ambiguities, so there is no need for the student network to learn them. On the other hand, a value of 1 indicates that the current pixel is the difficult pixel that the teacher has mastered while the student has not. This part of the pixels is the one that the student network is expected to learn well. The method of selecting pixels at the current stage is analogous to the teacher's highlighting operation in human education: most students usually cannot surpass the teacher's level, and if they expect to reach the teacher's level, then they only need to enhance their learning of the parts of the content that the student has not learned well, but the teacher has learned well.

After that, we apply $\mathrm{RD}_{\mathrm{TSE}}$ to L_{task} to get the total loss of the current stage, which is calculated as

$$L_{\mathrm{RDD}} = L_{\mathrm{TSE\text{-}RDD}} + L_{\mathrm{kd}} = \mathrm{RD}_{\mathrm{TSE}} \cdot L_{\mathrm{task}} + L_{\mathrm{kd}}. \tag{10}$$

By adjusting the loss weight of each pixel using $\mathrm{RD}_{\mathrm{TSE}}$, valuable difficult-to-classify pixels (with a corresponding $\mathrm{RD}_{\mathrm{TSE}}$ value of 1) receive more attention during training.

Algorithm 1 provides the pseudo-code illustrating the overall training pipeline of RDD. The proposed RDD learning paradigm leverages the teacher and student networks to deliver the relative difficulty of pixels to guide the network's learning.

3.6 Integrating with other approaches

Since RDD only affects the L_{task} loss, it can be seamlessly integrated with other methods without introducing additional optimization objectives. In this subsection, we demonstrate how to incorporate RDD into AT [52], DSD [74], and CIRKD [29] with multiple distillation losses, deriving the corresponding distillation loss formulations. To ensure consistency with the comparison results of other experiments

Algorithm 1 RDD

Input: Input images x, labels y, the parameter of teacher network θ_t, the iterations of TFE-RDD $\text{iter}_{\text{TFE-RDD}}$, cross-entropy loss function CE, kullback-leibler divergence function KL, softmax function σ.
Output: The parameter of student network θ_s.

while the student network has not converged **do**
  $L_{\text{task}} = \frac{1}{H \times W} \sum_{h=1}^{H} \sum_{w=1}^{W} \text{CE}\left(\sigma\left(\boldsymbol{Z}_{h,w}\right), \boldsymbol{y}_{h,w}\right)$;
  $L_{\text{kd}} = \frac{1}{H \times W} \sum_{h=1}^{H} \sum_{w=1}^{W} \text{KL}(\sigma(\frac{\boldsymbol{Z}^s_{h,w}}{T}) \| \sigma(\frac{\boldsymbol{Z}^t_{h,w}}{T}))$;
  if iter $\leqslant$ $\text{iter}_{\text{TFE-RDD}}$ **then**
    $D_{\text{kl}} = f(x \mid \theta_t) \log(\frac{f(x|\theta_t)}{f_{\text{aux}}(x|\theta_t)})$;
    $\text{RD}_{\text{TFE}} = \exp\{-D_{\text{kl}}\}$;
    $L_{\text{RDD}} = \text{RD}_{\text{TFE}} \cdot L_{\text{task}} + L_{\text{kd}}$;
  else
    $\text{RD}_{\text{TSE}} = (f(x \mid \theta_s) \leqslant t) \oplus (f(x \mid \theta_t) \leqslant t)$;
    $L_{\text{RDD}} = \text{RD}_{\text{TSE}} \cdot L_{\text{task}} + L_{\text{kd}}$;
  end if
  L_{overall}.backward();
end while
return θ_s.

and to conveniently integrate with other methods, we keep the hyperparameters in the integrated loss function consistent with those of the comparison method. Specific hyperparameter values are provided in each subsection below.

3.6.1 *The distillation loss of AT method after integrating RDD*

The total loss defined by AT [52]:

$$L_{\text{AT}} = L_{\text{task}} + \frac{\beta}{2} \sum_{j \epsilon I} \left\| \frac{Q_S^j}{||Q_S^j||_2} - \frac{Q_T^j}{||Q_T^j||_2} \right\|_2, \tag{11}$$

where Q_S^j and Q_T^j are respectively the j-th pair of student and teacher attention maps in vectorized form, $\frac{Q_S^j}{||Q_S^j||_2}$ and $\frac{Q_T^j}{||Q_T^j||_2}$ are the result of using l_2-normalization attention maps. The calculation details of $\frac{\beta}{2} \sum_{j \epsilon I} ||\frac{\boldsymbol{Q}_S^j}{||\boldsymbol{Q}_S^j||_2} - \frac{\boldsymbol{Q}_T^j}{||\boldsymbol{Q}_T^j||_2}||_2$ are described in AT [52]. L_{task} is the semantic segmentation task loss function mentioned in Subsection 3.1, represented by the cross entropy function.

The distillation loss of RDD:

$$L_{\text{RDD}} = \begin{cases} \text{RD}_{\text{TFE}} \cdot L_{\text{task}} + L_{\text{kd}}, & \text{iter} \leqslant \text{iter}_{\text{TFE-RDD}}, \\ \text{RD}_{\text{TSE}} \cdot L_{\text{task}} + L_{\text{kd}}, & \text{iter} > \text{iter}_{\text{TFE-RDD}}, \end{cases} \tag{12}$$

where RD_{TFE} represents the relative difficulty generated by the teacher network, and RD_{TSE} denotes the relative difficulty generated by the student-teacher cooperation. The calculations of RD_{TFE} and RD_{TSE} are described in Algorithm 1 in our paper. $\text{iter}_{\text{TFE-RDD}}$ is the iteration number of the TFE-RDD stage. L_{kd} is the pixel-level distillation loss mentioned in Subsection 3.1, represented by KL divergence.

The distillation loss of AT [52] method after integrating RDD can be derived

$$L_{\text{AT\_RDD}} = \begin{cases} \text{RD}_{\text{TFE}} \cdot L_{\text{task}} + \frac{\beta}{2} \sum_{j \epsilon I} \left\| \frac{Q_S^j}{||Q_S^j||_2} - \frac{Q_T^j}{||Q_T^j||_2} \right\|_2, & \text{iter} \leqslant \text{iter}_{\text{TFE-RDD}}, \\ \text{RD}_{\text{TSE}} \cdot L_{\text{task}} + \frac{\beta}{2} \sum_{j \epsilon I} \left\| \frac{Q_S^j}{||Q_S^j||_2} - \frac{Q_T^j}{||Q_T^j||_2} \right\|_2, & \text{iter} > \text{iter}_{\text{TFE-RDD}}. \end{cases} \tag{13}$$

In the new loss term described above, we follow the default parameter settings in AT [52] and set the weighting parameter β to 10^3 divided by the number of elements in the attention map and batch size for each layer. We only modify the weights for each pixel by incorporating the relative difficulty factors obtained through RDD in front of the L_{task}.

3.6.2 *The distillation loss of DSD method after integrating RDD*

The total loss defined by DSD [74]:

$$L_{\text{DSD}} = L_{\text{task}} + \alpha L_{\text{PSD}} + \beta L_{\text{CSD}}, \tag{14}$$

where L_{PSD} refers to the pixel-wise similarity distillation loss between the teacher and student networks, L_{CSD} refers to the category-wise similarity distillation loss between the teacher and student networks, and α and β are the weight balance parameters. The calculation details of L_{PSD} and L_{CSD} are described in DSD [74].

The distillation loss of RDD:

$$L_{\mathrm{RDD}} = \begin{cases} \mathrm{RD}_{\mathrm{TFE}} \cdot L_{\mathrm{task}} + L_{\mathrm{kd}}, & \mathrm{iter} \leqslant \mathrm{iter}_{\text{TFE-RDD}}, \\ \mathrm{RD}_{\mathrm{TSE}} \cdot L_{\mathrm{task}} + L_{\mathrm{kd}}, & \mathrm{iter} > \mathrm{iter}_{\text{TFE-RDD}}. \end{cases} \tag{15}$$

Hence, we derive the modified distillation loss for the DSD method after integrating RDD:

$$L_{\mathrm{DSD\_RDD}} = \begin{cases} \mathrm{RD}_{\mathrm{TFE}} \cdot L_{\mathrm{task}} + \alpha L_{\mathrm{PSD}} + \beta L_{\mathrm{CSD}}, & \mathrm{iter} \leqslant \mathrm{iter}_{\text{TFE-RDD}}, \\ \mathrm{RD}_{\mathrm{TSE}} \cdot L_{\mathrm{task}} + \alpha L_{\mathrm{PSD}} + \beta L_{\mathrm{CSD}}, & \mathrm{iter} > \mathrm{iter}_{\text{TFE-RDD}}. \end{cases} \tag{16}$$

In the new loss term described above, we follow the default parameter settings in DSD [74], setting the weighting parameter α to 10^3 and β to 10. It allows the student network to prioritize learning pixels that hold greater value at a given stage. Introducing this bias towards valuable pixel learning enhances the student network's overall performance.

3.6.3 *The distillation loss of CIRKD method after integrating RDD*

The total loss defined by CIRKD [29]:

$$L_{\mathrm{CIRKD}} = L_{\mathrm{task}} + L_{\mathrm{kd}} + \alpha L_{\mathrm{batch\_p2p}} + \beta L_{\mathrm{memory\_p2p}} + \gamma L_{\mathrm{memory\_p2r}}, \tag{17}$$

where $L_{\mathrm{batch\_p2p}}$ represents distillation loss of mini-batch-based pixel-to-pixel, $L_{\mathrm{memory\_p2p}}$ denotes distillation loss of memory-based pixel-to-pixel, $L_{\mathrm{memory\_p2r}}$ denotes distillation loss of memory-based pixel-to-region. α, β and γ are the weight balance parameters. The further calculation details of $L_{\mathrm{batch\_p2p}}$, $L_{\mathrm{memory\_p2p}}$ and $L_{\mathrm{memory\_p2r}}$ are described in [29].

The distillation loss of RDD:

$$L_{\mathrm{RDD}} = \begin{cases} \mathrm{RD}_{\mathrm{TFE}} \cdot L_{\mathrm{task}} + L_{\mathrm{kd}}, & \mathrm{iter} \leqslant \mathrm{iter}_{\text{TFE-RDD}}, \\ \mathrm{RD}_{\mathrm{TSE}} \cdot L_{\mathrm{task}} + L_{\mathrm{kd}}, & \mathrm{iter} > \mathrm{iter}_{\text{TFE-RDD}}. \end{cases} \tag{18}$$

After integrating RDD, the modified distillation loss of the CIRKD method is derived as follows:

$$L_{\mathrm{CIRKD\_RDD}} = \begin{cases} \mathrm{RD}_{\mathrm{TFE}} \cdot L_{\mathrm{task}} + L_{\mathrm{kd}} + \alpha L_{\mathrm{batch\_p2p}} + \beta L_{\mathrm{memory\_p2p}} + \gamma L_{\mathrm{memory\_p2r}}, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathrm{iter} \leqslant \mathrm{iter}_{\text{TFE-RDD}}, \\ \mathrm{RD}_{\mathrm{TSE}} \cdot L_{\mathrm{task}} + L_{\mathrm{kd}} + \alpha L_{\mathrm{batch\_p2p}} + \beta L_{\mathrm{memory\_p2p}} + \gamma L_{\mathrm{memory\_p2r}}, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathrm{iter} > \mathrm{iter}_{\text{TFE-RDD}}. \end{cases} \tag{19}$$

In this new loss term, we follow the default parameter settings in CIRKD [29], setting the weighting parameter α to 1, β to 0.1, and γ to 0.1. Moreover, RDD can seamlessly integrate with other semantic segmentation methods based on KD. This integration allows us to enhance the performance of the student network further from existing approaches.

4 Experiments

4.1 Data descriptions

We evaluate our method on the Cityscapes [75], CamVid [76], PASCAL VOC 2012 [77] and ADE20k [78]. Cityscapes [75] is a large-scale dataset for urban street scenes with 5000 high-quality images with pixel-wise annotations with 19 semantic classes. These finely annotated images are divided into 2975/500/1525 for train/val/test. CamVid [76] is an automotive driving dataset that contains 367/101/233 images for train/val/test with 11 semantic classes. Pascal VOC 2012 [77] is a general object segmentation benchmark with 21 classes. The original segmentation dataset is divided into 10582/1449/1456 for train/val/test. Following previous studies, we also use the extra annotations provided by [79]. ADE20k [78] is a scene segmentation dataset covering 150 fine-grained semantic classes with 20210 images.

4.2 Implementation details

Network architectures. We employ DeepLabV3 [17] with a ResNet-101 backbone [71] as the cumbersome teacher network for all experiments. As for student networks, we utilize various segmentation architectures to validate the effectiveness of our distillation methods. Specifically, we use DeepLabV3 [17] and PSPNet [20] with ResNet-18 backbone and MobileNetV2 [80] backbone for the student network.

Training details. The networks are trained using mini-batch stochastic gradient descent (SGD) with a momentum of 0.9 and weight decay of 0.0005. For Cityscapes, CamVid, and PASCAL VOC 2012 (VOC12), we set the number of iterations to 40000; for the ADE20k dataset, we set it to 80000. The learning rate is initialized at 0.02 and is multiplied by $(1 - \frac{\text{iter}}{\text{iter}_{\text{total}}})^{0.9}$ during training. We randomly crop images into sizes of 512×1024, 360×360, and 512×512 for Cityscapes, CamVid, Pascal VOC, and ADE20k datasets, respectively. Normal data augmentation techniques such as random flipping and scaling in the range of [0.5, 2] are applied during training. The threshold t of TSE-RDD in (9) is set as 0.7 (to be evaluated in the ablation study). The temperature T in L_{kd} is set to be 1. All experiments are conducted on four 3090 GPUs using mixed-precision training.

Evaluation metrics. Following the standard setting, we adopt the mIoU (mean intersection over union) metric to evaluate the performance of different methods. mIoU calculates the ratio of the intersection and union of two sets of true and predicted labels.

4.3 Comparison with existing methods on four datasets

In this subsection, we compare our proposed RDD with recent KD-based semantic segmentation methods, including SKD [28], IFVD [54], CWD [31], CIRKD [29], AT [52] and DSD [74] on four representative semantic segmentation datasets described above. The experimental results are shown in Tables 1–4 [22, 23, 25, 28–31, 52, 54, 74, 81–84]. In these tables, "T: DeepLabV3-Res101" denotes training with the teacher network. "S: DeepLabV3-Res18", "S: DeepLabV3-Res18*", "S: DeepLabV3-MBV2", and "S: PSPNet-Res18" denote training with the student networks. It should be noted that we used DeepLabV3-MBV2 as the student network, which uses the lightweight MobileNetV2 as the backbone network. This setup aims to validate the performance of RDD on lightweight networks in order to explore the possibility of deploying our approach on mobile devices.

(1) Results on cityscapes. To validate the performance of mIoU, we evaluate RDD on the Cityscapes [75] dataset. We chose the following non-KD semantic segmentation models as student networks: DeepLabV3 [17] based on ResNet-18 (Res18) backbone, DeepLabV3 [17] based on MobileNetV2 [80] backbone, and PSPNet [17] based on ResNet-18 (Res18) backbone. The experimental results are shown in Table 1. All structured KD methods are observed to improve the student network's segmentation performance compared to training without KD. Our proposed RDD achieves optimal performance across all four student networks, demonstrating its robustness to variations in student network architecture. Furthermore, for networks without ImageNet pre-training, RDD increases mIoU by 4.61%. This is mainly due to its progressive distillation design from easy to hard, which is more friendly to student networks with minimal knowledge. Figure 5 shows the qualitative segmentation results on the verification set using the DeepLabV3-ResNet18 network. The validity of our proposed approach is intuitively demonstrated, and the semantic labels produced by RDD are more consistent with the ground truth.

Like other semantic segmentation methods based on KD, RDD does not modify the model architecture. Therefore, the parameters (Params) of the model and the floating point operations (FLOPs) resulting from inference remain consistent with the underlying backbone network. Specifically, for all methods where the student model is DeepLabV3-Res18, the FLOPs are 572 G and the Params are 13.6 M. Similarly, for all methods where the student model is PSPNet-Res18, the FLOPs are 507.4 G, and the Params are 12.9 M. Additionally, we verify the performance of our method on lightweight networks, such as using MobileNetV2 as the student network. As can be seen from the table, although the model based on MobileNetV2 has fewer parameters and fewer calculations (128.9 G FLOPs, 3.2 M Params), its mIoU reaches 76.38%. It is even 0.5% mIoU higher than the model based on PSPNet-Res18 (507.4 G FLOPs, 12.9 M Params) with larger parameters and calculations. This shows that our method is also suitable for some specially designed lightweight models, such as SqueezeNet [82] series, MobileNet [80] series, and ShuffleNet [83, 84] series. RDD can achieve high segmentation accuracy while using as few computing resources and parameters as possible, thereby better deploying in resource-constrained environments such as embedded and mobile devices.

Table 1 Performance comparison with SOTA KD methods over various student segmentation networks on Cityscapes[a)]

Method	mIoU (%)	FLOPs (G)	Params (M)
Some related non-KD semantic segmentation methods			
ENet [22]	58.42	14.4	0.35
ESPNet [25]	60.34	17.7	0.36
ICNet [23]	69.53	113.2	26.5
FCN [82]	62.76	1335.6	134.5
RefineNet [83]	73.58	2102.8	118.1
OCNet [84]	80.12	2194.2	62.6
Comparison with different KD methods			
T: DeepLabV3-Res101	78.07	2371	61.1
S: DeepLabV3-Res18	74.21	572.0	13.6
+ SKD [28]	75.42 (↑ 1.21)		
+ IFVD [54]	75.59 (↑ 1.38)		
+ CWD [31]	75.55 (↑ 1.34)		
+ CIRKD [29]	76.38 (↑ 2.17)		
+ RDD (ours)	**77.18 (↑ 2.97)**		
S: DeepLabV3-Res18*	65.17	572.0	13.6
+ SKD [28]	67.08 (↑ 1.91)		
+ IFVD [54]	65.96 (↑ 0.79)		
+ CWD [31]	67.74 (↑ 2.57)		
+ CIRKD [29]	68.18 (↑ 3.01)		
+ RDD (ours)	**69.78 (↑ 4.61)**		
S: DeepLabV3-MBV2	73.12	128.9	3.2
+ SKD [28]	73.82 (↑ 0.70)		
+ IFVD [54]	73.50 (↑ 0.38)		
+ CWD [31]	74.66 (↑ 1.54)		
+ CIRKD [29]	75.42 (↑ 2.30)		
+ RDD (ours)	**76.38 (↑ 3.26)**		
S: PSPNet-Res18	72.55	507.4	12.9
+ SKD [28]	73.29 (↑ 0.74)		
+ IFVD [54]	73.71 (↑ 1.16)		
+ CWD [31]	74.36 (↑ 1.81)		
+ CIRKD [29]	74.73 (↑ 2.18)		
+ RDD (ours)	**75.88 (↑ 3.33)**		

a) FLOPs are measured based on the test size of 1024 × 2048. * denotes that we do not initialize the backbone with ImageNet [81] pre-trained weights. The bold parts are the best results.

In Table 1, we list several related non-KD semantic segmentation methods. On the one hand, lightweight models like ENet [22], ESPNet [25], and ICNet [23] require very little FLOPs. They are suitable for deployment on lightweight mobile devices but have relatively low mIoU. On the other hand, models like FCN [85], RefineNet [86], and OCRNet [87] have higher mIoU but are very computationally intensive and difficult to deploy on resource-constrained devices. RDD methods achieve a certain balance between computational efficiency and segmentation performance compared to non-KD methods. This makes the RDD an effective semantic segmentation method, more suitable for resource-constrained devices.

(2) Results on pascal VOC 2012. We also evaluate RDD on Pascal VOC 2012 [77], a representative visual object segmentation dataset. We use the DeepLabV3 [17] with the backbone of ResNet-18 (Res18) and PSPNet [20] with the backbone of ResNet-18 (Res18) as two student networks. As shown in Table 2, RDD outperforms other KD methods based on semantic segmentation. Compared to the original student networks, RDD improves mIoU by 1.67Specifically, for DeepLabV3 with ResNet-18 backbone, mIoU increased from 73.21% to 74.88%. For PSPNet with ResNet-18 backbone, mIoU improved from 73.33% to 74.83%.

(3) Results on ADE20k. Table 3 compares the performance of RDD with state-of-the-art distillation methods on the ADE20k dataset. We use the DeepLabV3 [17] with ResNet-18 (Res18) backbone as the student network. RDD achieves the best performance compared to standard KD [30], AT [52], and DSD [74]. For DeepLabV3 with ResNet-18 backbone, mIoU on the validation set increases from 36.52% to

Table 2 Performance comparison with SOTA KD methods over various student segmentation networks on Pascal VOC 2012[a)]

Method	mIoU (%)	FLOPs (G)	Params (M)
T: DeepLabV3-Res101	77.67	1294.6	61.1
S: DeepLabV3-Res18	73.21	305.0	13.6
+ SKD [28]	73.51 (↑ 0.30)		
+ IFVD [54]	73.85 (↑ 0.64)		
+ CWD [31]	74.02 (↑ 0.81)		
+ CIRKD [29]	74.50 (↑ 1.29)		
+ RDD (ours)	**74.88 (↑ 1.67)**		
S: PSPNet-Res18	73.33	260.0 G	12.9 M
+ SKD [28]	74.07 (↑ 0.74)		
+ IFVD [54]	73.54 (↑ 0.21)		
+ CWD [31]	73.99 (↑ 0.66)		
+ CIRKD [29]	74.78 (↑ 1.45)		
+ RDD (ours)	**74.82 (↑ 1.49)**		

a) FLOPs are measured based on the test size of 512 × 512. The bold parts are the best results.

Table 3 Performance comparison with SOTA KD methods over various student segmentation networks on ADE20k[a)]

Method	mIoU (%)	FLOPs (G)	Params (M)
T: DeepLabV3-Res101	42.70	1294.6	61.1
S: DeepLabV3-Res18	36.52	305.0	13.6
+ KD [30]	36.63 (↑ 0.11)		
+ AT [52]	37.79 (↑ 1.27)		
+ DSD [74]	37.37 (↑ 0.85)		
+ RDD (ours)	**38.04 (↑ 1.52)**		

a) FLOPs are measured based on the test size of 512 × 512. The bold part is the best results.

Table 4 Performance comparison with SOTA KD methods over various student segmentation networks on CamVid[a)]

Method	mIoU (%)	FLOPs (G)	Params (M)
T: DeepLabV3-Res101	69.84	280.2	61.1
S: DeepLabV3-Res18	66.92	61.0	13.6
+ SKD [28]	67.46 (↑ 0.54)		
+ IFVD [54]	67.28 (↑ 0.36)		
+ CWD [31]	67.71 (↑ 0.79)		
+ CIRKD [29]	68.21 (↑ 1.29)		
+ RDD (ours)	**68.55 (↑ 1.63)**		
S: PSPNet-Res18	66.73	45.6	12.9
+ SKD [28]	67.83 (↑ 1.10)		
+ IFVD [54]	67.61 (↑ 0.88)		
+ CWD [31]	67.92 (↑ 1.19)		
+ CIRKD [29]	68.65 (↑ 1.92)		
+ RDD (ours)	**68.77 (↑ 2.04)**		

a) FLOPs are measured based on the test size of 360 × 480. The bold parts are the best results.

38.04%. Our method based on the same network also outperforms CIRKD's result of 37.07%. The experimental results demonstrate the effectiveness and generality of our approach, and RDD achieves consistent performance improvement in different segmentation networks and further narrows the performance gap with teacher networks.

(4) Results on CamVid. We evaluate various distillation methods on CamVid (a simple small scene understanding dataset) dataset shown in Table 4. RDD consistently achieves optimal performance. We use the DeepLabV3 [17] with ResNet-18 (Res18) backbone and the PSPNet [20] with ResNet-18 (Res18) backbone as two student networks. Due to the simplicity of the dataset, the compact student network can obtain segmentation performance close to the cumbersome teacher network. However, RDD can still improve the upper bound of the performance of the student network. Compared to the original student networks, our method improves the mIoU of the two student networks by 1.63% and 2.04%, respectively. Specifically, for DeepLabV3 with ResNet-18 backbone, mIoU increased from 66.92% to 68.55%. For

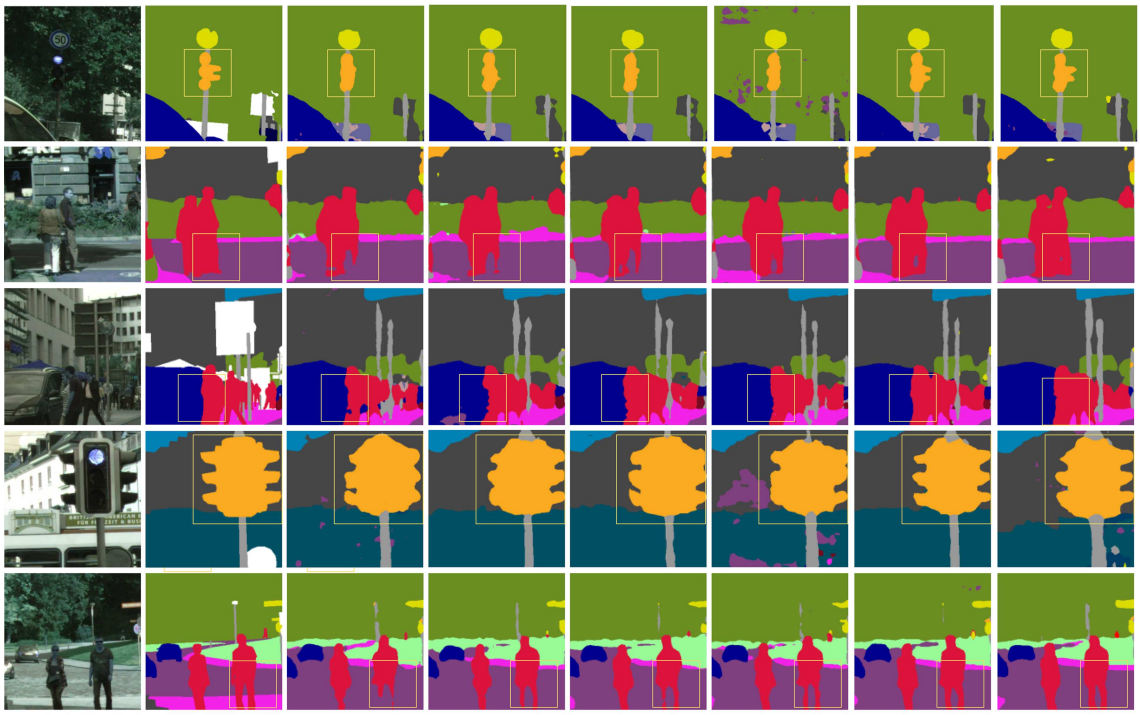

Figure 5 Qualitative segmentation results on the validation set of Cityscapes using DeepLabV3-ResNet18 as the student network and DeepLabV3-ResNet101 as the teacher network. (a) Input image; (b) ground truth; (c) results of the original student network without KD; (d) results of AT [52]; (e) results of CIRKD [29]; (f) results of DSD [74]; (g) results of the proposed RDD; (h) results of the teacher network.

PSPNet with ResNet-18 backbone, mIoU improved from 66.73% to 68.77%.

4.4 Ablation study

4.4.1 *Ablation experiment for the two-stage switch*

We conduct ablation experiments on the Cityscape dataset to evaluate the effectiveness of the two-stage switch in RDD. For these experiments, we employ the segmentation framework DeepLabV3 [17] with ResNet-101 (Res101) backbone [71] as the powerful teacher network and DeepLabV3 [17] with ResNet-18 (Res18) backbone as the student network. The ablation experimental results are summarized in Table 5. RDD consists of two stages: TFE-RDD and TSE-RDD. The experimental group (a) indicates that only TFE-RDD is involved throughout the whole training process, and the experimental group (b) indicates that only TSE-RDD is involved in the whole training process. Experimental groups (c)–(e) investigate the role played by TFE-RDD and the proportion's effect of training phase iterations accounted for by TE-REDD on the performance of the student network.

We adopted warm-up training in the early learning stage. The teacher network provides supervision information to guide the student network to learn relatively simple pixels and achieve rapid convergence. This stage of training should be short to avoid a warm-up period that is too long and cannot give full play to the potential of the student model itself. To determine the optimal percentage parameter p of training iterations, we tried four different percentages: 0%, 10%, 20%, 30%. By evaluating the performance of the student network at different scales, we aim to find the most suitable parameter p to achieve a balance between fast learning and giving full play to the potential of the student network. The group (c) indicates that the percentage of TFE-RDD stage iterations is 0%, the same as the experimental group (b).

The experimental results for groups (a) and (b) indicate that the TSE-RDD stage significantly affects the network performance. Without TSE-RDD, the performance is almost the same as the baseline, and the network only has an improvement of 0.22% mIoU. According to the analysis, without the TSE-RDD stage involved in training, the network consistently tends to focus primarily on learning simple pixels. It keeps the learning preference for difficult pixels at a low level. This phenomenon is similar to the comfort zone in human education, where too much simple training does not easily raise the learner's

Table 5 Effect of components in the proposed method[a)]

	Distillation		Training	mIoU (%)
	TFE-RDD	TSE-RDD	Training iterations p	
T: DeepLabV3-Res101				78.07
S: DeepLabV3-Res18				74.21
(a)	✓			74.43 (↑ 0.22)
(b)		✓		76.34 (↑ 2.13)
(c)	✓	✓	0%	76.34 (↑ 2.13)
(d)	✓	✓	10%	**77.18 (↑ 2.97)**
(e)	✓	✓	20%	76.77 (↑ 2.56)
(f)	✓	✓	30%	76.58 (↑ 2.37)

a) The bold part is the best results.

Table 6 Effect of combination mode of teacher and student difficulty maps in the TSE-RDD stage[a)]

	Distillation		Training	mIoU (%)
	TFE-RDD	TSE-RDD	Training iterations p	
T: DeepLabV3-Res101				78.07
S: DeepLabV3-Res18				74.21
AND	✓	✓	10%	70.01
OR	✓	✓	10%	75.32
XOR	✓	✓	10%	**77.18**

a) The bold part is the best results.

upper performance bound. Experimental groups (d), (e), and (f) evaluated the contribution of the TFE-RDD stage. The results suggest that the TFE-RDD stage is the icing on the cake, and the performance is further improved. Through the analysis, we find that in the early learning stage, the student network has not yet converged, and the judgment of difficult pixels is not stabilizable, so the direct application of TSE-RDD does not work well. The experimental results show that applying TFE-RDD during the initial 10% of training iterations and TSE-RDD in the remaining iterations yields the best KD performance.

4.4.2 *Ablation experiment of TSE-RDD mode*

TSE-RDD's combination of the teacher and student networks can be expressed as different bitwise operations, including XOR, AND, and OR operations. In our TSE-RDD stage, the relative difficulty $\mathrm{RD}_{\mathrm{TS}}$ obtained by the XOR operation between the teacher and the student networks is expressed as $\mathrm{RD}_{\mathrm{TSE}} = (f(x \mid \theta_s) \leqslant t) \oplus (f(x \mid \theta_t) \leqslant t)$. The strategy focuses on selecting pixels where the teacher network's predictions are above threshold t while the student network's predictions are below threshold t. It encourages the student network to learn the difficult pixels that the teacher network has mastered and the student network has not yet mastered but is capable of learning. Similarly, when using an AND operation between the teacher and student networks, we obtain $\mathrm{RD}_{\mathrm{TSE}} = (f(x \mid \theta_s) \leqslant t) \wedge (f(x \mid \theta_t) \leqslant t)$. This strategy selects pixels where the prediction confidences of both the teacher and student networks are below threshold t. It encourages the student network to learn parts of pixels that are difficult for even the teacher network to master. Lastly, we consider an OR operation between them: $\mathrm{RD}_{\mathrm{TSE}} = (f(x \mid \theta_s) \leqslant t) \vee (f(x \mid \theta_t) \leqslant t)$. This strategy combines results obtained from both XOR and AND operations. It encourages the student network to learn difficult pixels that promise to be learned well and difficult pixels that even the teacher network struggles to master.

We conduct ablation experiments on the influence of these three bitwise operation modes on our TSE-RDD stage. The experimental results are shown in Table 6. The OR mode's mIoU decreased by 1.86% compared to the XOR mode. Analysis reveals that compared to XOR mode, OR mode also introduces pixels that are difficult even for the teacher network to master, and these pixels can be difficult pixels with noise annotations or semantic ambiguities. These challenging pixels can negatively impact network performance. The AND mode achieved mIoU of 70.01%, which is 4.20% lower than the student network without the KD method. Analysis reveals that AND mode introduces too many difficult pixels for the student network and only encourages students to learn difficult pixels that even the teacher network struggles to master. The AND mode's experimental results indicate that using excessively difficult pixels only in the later stages of training would severely hinder student network learning. In contrast, the OR mode not only discards the simple pixels that the student network has already mastered and does not

Table 7 Effect of threshold t in the TSE-RDD stage[a)]

	$t = 0.60$	$t = 0.65$	$t = 0.70$	$t = 0.75$	$t = 0.80$
mIoU (%)	76.47	76.88	**76.95**	76.66	76.32

a) The bold part is the best results.

need to learn but also separates difficult pixels from those that may be noisy annotations or have semantic ambiguities. It selectively learns the most valuable difficult pixels that the student network is expected to master. As a result, the OR mode achieves the best performance.

4.4.3 *Ablation experiments with threshold in TSE-RDD*

We investigate the effect of threshold t on determining the number of difficult pixels to retain in the TSE-RDD stage. In the MMSegmentation code base [88], the confidence threshold used to filter difficult samples is set to 0.7. To further investigate the impact of threshold t, we performed a set of ablation experiments. In the experiments, we use the DeepLabV3 [17] with ResNet-101 (Res101) backbone [71] as the teacher network and DeepLabV3 [17] with ResNet-18 (Res18) backbone as the student network. The experimental results are presented in Table 7 and averaged over three independent runs. It can be observed that an appropriate threshold t is effective in providing an optimal number of difficult pixels, leading to improved learning for the student network. However, setting a threshold t that is too large or too small has a negative impact on the student network's learning. Analysis reveals that setting threshold t too small would discard too many relatively difficult pixels, thus failing to enhance the student network's upper performance bound. Conversely, when threshold t is set excessively large, many simple samples will be retained. It makes it difficult to play out the learning that TSE-RDD would have done by utilizing difficult pixels. The experimental results show that the RDD method performs best when $t = 0.70$.

4.5 Integrating with other approaches and training time

We evaluate integrating RDD's impact into existing KD approaches, including AT [52], DSD [74], CIRKD [29], and KD [30]. The KD group only uses the spatial distillations shown in (2). In these methods, we use DeepLabV3-Res18 as the student model. When integrating with other methods, we did not change any hyperparameters under the original method's experimental settings to explore our method's effectiveness in the simplest integration mode. The experimental results are presented in Table 8. In the four baseline approaches, RDD effectively improves the performance of all approaches, further narrowing the performance gap between the student and teacher networks. After integrating RDD, mIoU for each approach shows improvement. Specifically, the AT method's mIoU after integrating RDD increases by 0.9%. DSD method's mIoU after integrating RDD increases by 1.3%. CIRKD method's mIoU after integrating RDD increases by 0.31%. For the KD method, the mIoU increases by 2.36% after integration. Figure 6 compares prediction maps for methods with and without RDD integration. The visualization results of the methods integrated with RDD generally perform better on complex regions such as edges and obscured objects.

Furthermore, we access the training time before and after integrating various KD methods with RDD. Table 9 displays the experimental results comparing training times. As can be seen from the table, among the many KD methods, RDD requires the shortest training time, which is only 4 h and 20 min. In comparison, the training time of AT, CIRKD, and DSD methods is longer than RDD. It is worth noting that the CIRKD method employs a memory bank for contrastive learning and needs to optimize five loss terms, which results in its training time of up to 7 h and 38 min, almost twice as long as other methods. This also verifies the problem that adding multiple optimization objectives will increase training difficulty and time. This comparison concludes that integrating RDD does not incur excessive computational overhead or training time. It only incurs an average increase of less than 10% in training time across different methods.

5 Conclusion and future work

We propose RDD for semantic segmentation. Unlike previous feature- and response-based distillation methods, we leverage the relative difficulty of samples as dark knowledge transfers from teacher to student and adjust the learning procedure based on it. We propose TFE-RDD and TSE-RDD for use at different

Table 8 Integrating with other KD approaches[a)]

Method	mIoU (%)	FLOPs (G)	Params (M)
T: DeepLabV3-Res101	78.07	2371	61.1
S: DeepLabV3-Res18	74.21	572.0	13.6
+ AT [52]	76.21	572.0	13.6
+ AT [52] + RDD (ours)	**77.11** (↑ 0.90)		
+ DSD [74]	74.42	572.0	13.6
+ DSD [74] + RDD (ours)	**75.72** (↑ 1.30)		
+ CIRKD [29]	76.38	572.0	13.6
+ CIRKD [29] + RDD (ours)	**76.69** (↑ 0.31)		
+ KD [30]	74.82	572.0	13.6
+ KD [30] + RDD (ours)	**77.18** (↑ 2.36)		

a) "+" denotes implementing the corresponding schemes. The bold parts are the best results.

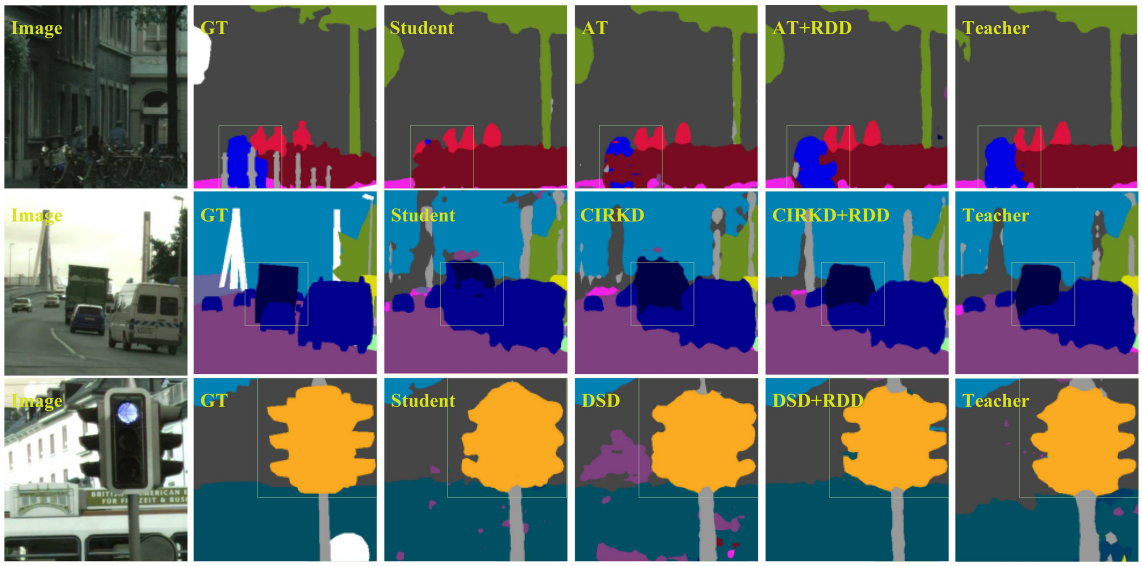

Figure 6 Predictions of KD methods with and without integrating RDD.

Table 9 Comparison of training time for experiments with and without RDD integration

Method	Cost	Method with RDD	Cost
Baseline	4 h 05 min	Baseline with RDD	4 h 20 min
AT [52]	4 h 30 min	AT [50] with RDD	4 h 52 min
CIRKD [29]	7 h 38 min	CIRKD [29] with RDD	8 h 02 min
DSD [74]	4 h 32 min	DSD [74] with RDD	4 h 56 min

stages of student network training. By tending to learn simpler pixels in the early stage to achieve rapid convergence and focusing on valuable difficult pixels in the later stage to enhance the performance ceiling, the student network effectively improves its overall performance. RDD is easy to integrate with existing distillation methods. Experimental results demonstrate RDD outperforms state-of-the-art KD methods.

In this work, we focus on the RDD of the semantic segmentation task. However, integrating RDD into other vision tasks, such as classification and detection, is also feasible. In future research, we will further explore other potential effects of the relative difficulty of KD and explore adaptability in tasks beyond semantic segmentation.

Acknowledgements This work was partly supported by National Natural Science Foundation of China (Grant Nos. 62272229, 62076124, 62222605), National Key R&D Program of China (Grant No. 2020AAA0107000), Natural Science Foundation of Jiangsu Province (Grant Nos. BK20222012, BK20211517), and Shenzhen Science and Technology Program (Grant No. JCYJ2023080714200-1004). The authors would like to thank all the anonymous reviewers for their constructive comments.

References

1 Gare G R, Li J, Joshi R, et al. W-Net: dense and diagnostic semantic segmentation of subcutaneous and breast tissue in ultrasound images by incorporating ultrasound RF waveform data. Med Image Anal, 2022, 76: 102326

2 Hu K, Zhang Z, Niu X, et al. Retinal vessel segmentation of color fundus images using multiscale convolutional neural network with an improved cross-entropy loss function. Neurocomputing, 2018, 309: 179–191
3 Kamnitsas K, Ledig C, Newcombe V F J, et al. Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. Med Image Anal, 2017, 36: 61–78
4 Kar M K, Nath M K, Neog D R. A review on progress in semantic image segmentation and its application to medical images. SN Comput Sci, 2021, 2: 397
5 Seidlitz S, Sellner J, Odenthal J, et al. Robust deep learning-based semantic organ segmentation in hyperspectral images. Med Image Anal, 2022, 80: 102488
6 Alonso I, Riazuelo L, Murillo A C. MiniNet: an efficient semantic segmentation ConvNet for real-time robotic applications. IEEE Trans Robot, 2020, 36: 1340–1347
7 Milioto A, Lottes P, Stachniss C. Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in CNNs. In: Proceedings of the IEEE International Conference on Robotics and Automation, 2018. 2229–2235
8 Nilsson D, Pirinen A, Gärtner E, et al. Embodied visual active learning for semantic segmentation. AAAI, 2021, 35: 2373–2383
9 Sun Y, Pan B, Fu Y. Lightweight deep neural network for real-time instrument semantic segmentation in robot assisted minimally invasive surgery. IEEE Robot Autom Lett, 2021, 6: 3870–3877
10 Zurbrugg R, Blum H, Cadena C, et al. Embodied active domain adaptation for semantic segmentation via informative path planning. IEEE Robot Autom Lett, 2022, 7: 8691–8698
11 Cui H, Radosavljevic V, Chou F C, et al. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In: Proceedings of the IEEE International Conference on Robotics and Automation, 2019. 2090–2096
12 Feng D, Haase-Schutz C, Rosenbaum L, et al. Deep multi-modal object detection and semantic segmentation for autonomous driving: datasets, methods, and challenges. IEEE Trans Intell Transp Syst, 2020, 22: 1341–1360
13 Menze M, Geiger A. Object scene flow for autonomous vehicles. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. 3061–3070
14 Siam M, Elkerdawy S, Jagersand M, et al. Deep semantic segmentation for automated driving: taxonomy, roadmap and challenges. In: Proceedings of the IEEE Conference on Intelligent Transportation Systems, 2017. 1–8
15 Treml M, Arjona M J. Speeding up semantic segmentation for autonomous driving. In: Proceedings of the Advances in Neural Information Processing Systems, 2016
16 Chen L C, Papandreou G, Kokkinos I, et al. DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans Pattern Anal Mach Intell, 2017, 40: 834–848
17 Chen L C, Papandreou G, Schroff F, et al. Rethinking atrous convolution for semantic image segmentation. 2017. ArXiv:1706.05587
18 Chen L C, Zhu Y, Papandreou G, et al. Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision, 2018. 801–818
19 Chen L C Papandreou G, Kokkinos I, et al. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In: Proceedings of the International Conference on Learning Representations, 2015
20 Zhao H, Shi J, Qi X, et al. Pyramid scene parsing network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017. 2881–2890
21 Wang J, Sun K, Cheng T, et al. Deep high-resolution representation learning for visual recognition. IEEE Trans Pattern Anal Mach Intell, 2020, 43: 3349–3364
22 Paszke A, Chaurasia A, Kim S, et al. ENet: a deep neural network architecture for real-time semantic segmentation. 2016. ArXiv:1606.02147
23 Zhao H, Qi X, Shen X, et al. ICNet for real-time semantic segmentation on high-resolution images. In: Proceedings of the European Conference on computer vision, 2018. 405–420
24 Yu C, Wang J, Peng C, et al. BiSeNet: bilateral segmentation network for real-time semantic segmentation. In: Proceedings of the European Conference on computer vision, 2018. 325–341
25 Mehta S, Rastegari M, Caspi A, et al. ESPNet: efficient spatial pyramid of dilated convolutions for semantic segmentation. In: Proceedings of the European Conference on computer vision, 2018. 552–568
26 Wu J, Leng C, Wang Y, et al. Quantized convolutional neural networks for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. 4820–4828
27 He W, Wu M, Liang M, et al. CAP: context-aware pruning for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021. 960–969
28 Liu Y, Chen K, Liu C, et al. Structured knowledge distillation for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 2604–2613
29 Yang C, Zhou H, An Z, et al. Cross-image relational knowledge distillation for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. 12319–12328
30 Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. 2015. ArXiv:1503.02531
31 Shu C, Liu Y, Gao J, et al. Channel-wise knowledge distillation for dense prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021. 5311–5320
32 Kendall A, Gal Y, Cipolla R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 7482–7491
33 Yu T, Kumar S, Gupta A, et al. Gradient surgery for multi-task learning. In: Proceedings of the International Conference on Neural Information Processing Systems, 2020. 5824–5836
34 Brophy J. Teacher influences on student achievement. Am Psychologist, 1986, 41: 1069–1077
35 Shrivastava A, Gupta A, Girshick R. Training region-based object detectors with online hard example mining. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. 761–769
36 Li H, Lin Z, Shen X, et al. A convolutional neural network cascade for face detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. 5325–5334
37 Nie D, Wang L, Xiang L, et al. Difficulty-aware attention network with confidence learning for medical image segmentation. AAAI, 2019, 33: 1085–1092
38 Carnine D, Silbert J, Kameenui E J, et al. Direct Instruction Reading. Columbus: Merrill, 1997
39 Bruner J S. Toward a Theory of Instruction. Cambridge: Harvard University Press, 1966
40 Midgley C. Goals, Goal Structures, and Patterns of Adaptive Learning. Abingdon: Routledge, 2014
41 Vogt F, Rogalla M. Developing adaptive teaching competency through coaching. Teach Teacher Educ, 2009, 25: 1051–1060

42 Grant P, Basye D. Personalized Learning: A Guide for Engaging Students with Technology. Arlington: International Society for Technology in Education, 2014
43 Zhou Z H, Jiang Y. NeC4.5: neural ensemble based C4.5. IEEE Trans Knowl Data Eng, 2004, 16: 770–773
44 Buciluǎ C, Caruana R, Niculescu-Mizil A. Model compression. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006. 535–541
45 Huang Z, Wang N. Like what you like: knowledge distill via neuron selectivity transfer. 2017. ArXiv:1707.01219
46 Peng B, Jin X, Liu J, et al. Correlation congruence for knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 5007–5016
47 Romero A, Ballas N, Kahou S E, et al. Fitnets: hints for thin deep nets. 2014. ArXiv:1412.6550
48 Tung F, Mori G. Similarity-preserving knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 1365–1374
49 Xu Z, Hsu Y C, Huang J. Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks. 2017. ArXiv:1709.00513
50 Yang C, An Z, Cai L, et al. Mutual contrastive learning for visual representation learning. AAAI, 2022, 36: 3045–3053
51 Yang C, An Z, Xu Y. Multi-view contrastive learning for online knowledge distillation. In: Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing, 2021. 3750–3754
52 Komodakis N, Zagoruyko S. Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer. In: Proceedings of the International Conference on Learning Representations, 2017
53 Liang D, Du Y, Sun H, et al. NLKD: using coarse annotations for semantic segmentation based on knowledge distillation. In: Proceedings of the IEEE Conference on Acoustics Speech and Signal Processing (ICASSP), 2021. 2335–2339
54 Wang Y, Zhou W, Jiang T, et al. Intra-class feature variation distillation for semantic segmentation. In: Proceedings of the European Conference on Computer Vision, 2020. 346–362
55 Zheng Z, Yang Y. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. Int J Comput Vis, 2021, 129: 1106–1120
56 Holder C J, Shafique M. Efficient uncertainty estimation in semantic segmentation via distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021. 3087–3094
57 Ji D, Wang H, Tao M, et al. Structural and statistical texture knowledge distillation for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022. 16876–16885
58 Bengio Y, Louradour J, Collobert R, et al. Curriculum learning. In: Proceedings of the International Conference on Machine Learning, 2009. 41–48
59 Jiang L, Meng D, Yu S I, et al. Self-paced learning with diversity. In: Proceedings of the International Conference on Neural Information Processing Systems, 2014
60 Ying W, Zhang Y, Huang J, et al. Transfer learning via learning to transfer. In: Proceedings of the International Conference on Machine Learning, 2018. 5085–5094
61 Wang C, Yang K, Zhang S, et al. TC3KD: knowledge distillation via teacher-student cooperative curriculum customization. Neurocomputing, 2022, 508: 284–292
62 Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, 2017. 2980–2988
63 Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci, 1997, 55: 119–139
64 Cao Y, Chen K, Loy C C, et al. Prime sample attention in object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020. 11583–11591
65 Li B, Liu Y, Wang X. Gradient harmonized single-stage detector. AAAI, 2019, 33: 8577–8584
66 Luo Y, Liu P, Guan T, et al. Significance-aware information bottleneck for domain adaptive semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 6778–6787
67 Tsai Y H, Hung W C, Schulter S, et al. Learning to adapt structured output space for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 7472–7481
68 Tsai Y H, Sohn K, Schulter S, et al. Domain adaptation for structured output via discriminative patch representations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 1456–1465
69 Kendall A, Gal Y. What uncertainties do we need in Bayesian deep learning for computer vision? In: Proceedings of the International Conference on Neural Information Processing Systems, 2017
70 Teye M, Azizpour H, Smith K. Bayesian uncertainty estimation for batch normalized deep networks. In: Proceedings of the International Conference on Machine Learning, 2018. 4907–4916
71 He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. 770–778
72 Du Y, Liang D, Quan R, et al. More than accuracy: an empirical study of consistency between performance and interpretability. In: Proceedings of the Pacific Rim International Conference on Artificial Intelligence, 2022. 579–590
73 Huang J, Qu L, Jia R, et al. O2U-Net: a simple noisy label detection approach for deep neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 3326–3334
74 Feng Y, Sun X, Diao W, et al. Double similarity distillation for semantic image segmentation. IEEE Trans Image Process, 2021, 30: 5363–5376
75 Cordts M, Omran M, Ramos S, et al. The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. 3213–3223
76 Brostow G J, Fauqueur J, Cipolla R. Semantic object classes in video: a high-definition ground truth database. Pattern Recogn Lett, 2009, 30: 88–97
77 Everingham M, Van Gool L, Williams C K I, et al. The Pascal visual object classes (VOC) challenge. Int J Comput Vis, 2010, 88: 303–338
78 Zhou B, Zhao H, Puig X, et al. Scene parsing through ADE20k dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017. 633–641
79 Hariharan B, Arbeláez P, Bourdev L, et al. Semantic contours from inverse detectors. In: Proceedings of the IEEE International Conference on Computer Vision, 2011. 991–998
80 Sandler M, Howard A, Zhu M, et al. MobileNetV2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 4510–4520
81 Russakovsky O, Deng J, Su H, et al. Imagenet large scale visual recognition challenge. Int J Comput Vis, 2015, 115: 211–252

82 Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: alexnet-level accuracy with 50x fewer parameters and < 0.5 MB model size. 2016. ArXiv:1602.07360
83 Zhang X, Zhou X, Lin M, et al. ShuffleNet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018. 6848–6856
84 Ma N, Zhang X, Zheng H T, et al. ShuffleNet V2: practical guidelines for efficient CNN architecture design. In: Proceedings of the European Conference on Computer Vision, 2018. 116–131
85 Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. 3431–3440
86 Lin G, Milan A, Shen C, et al. RefineNet: multi-path refinement networks for high-resolution semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017. 1925–1934
87 Yuan Y, Huang L, Guo J, et al. OCNet: object context network for scene parsing. 2018. ArXiv:1809.00916
88 Contributors M M S. MMSegmentation: OpenMMLab semantic segmentation toolbox and benchmark. 2020. https://github.com/open-mmlab/mmsegmentation