Endpoints Weight Fusion for Class Incremental Semantic Segmentation

Jia-Wen Xiao [1*] Chang-Bin Zhang [1*] Jiekang Feng [2] Xialei Liu [1†]
Joost van de Weijer [3] Ming-Ming Cheng [1]
[1] VCIP, CS, Nankai University [2] Tianjin University
[3] Computer Vision Center, Universitat Autònoma de Barcelona

Abstract

Class incremental semantic segmentation (CISS) focuses on alleviating catastrophic forgetting to improve discrimination. Previous work mainly exploits regularization (e.g., knowledge distillation) to maintain previous knowledge in the current model. However, distillation alone often yields limited gain to the model since only the representations of old and new models are restricted to be consistent. In this paper, we propose a simple yet effective method to obtain a model with a strong memory of old knowledge, named Endpoints Weight Fusion (EWF). In our method, the model containing old knowledge is fused with the model retaining new knowledge in a dynamic fusion manner, strengthening the memory of old classes in ever-changing distributions. In addition, we analyze the relationship between our fusion strategy and a popular moving average technique EMA, which reveals why our method is more suitable for class-incremental learning. To facilitate parameter fusion with closer distance in the parameter space, we use distillation to enhance the optimization process. Furthermore, we conduct experiments on two widely used datasets, achieving state-of-the-art performance.

1. Introduction

As a fundamental task, semantic segmentation plays a key role in visual applications [10, 25]. Previous fully-supervised works aim to segment fixed classes defined in the training set. However, the trained segmentation model is expected to recognize more classes in realistic applications. One straightforward solution is to re-train the model on the entire dataset by mixing old and new data. Nevertheless, this strategy will bring huge labeling and training costs. From the transfer learning perspective [22, 30], another plain solution is to adjust the previously learned model on the newly added data. But the model will overfit to new classes quickly, while forgetting previous old classes. This phenomenon is also known as *catastrophic forgetting* [35].

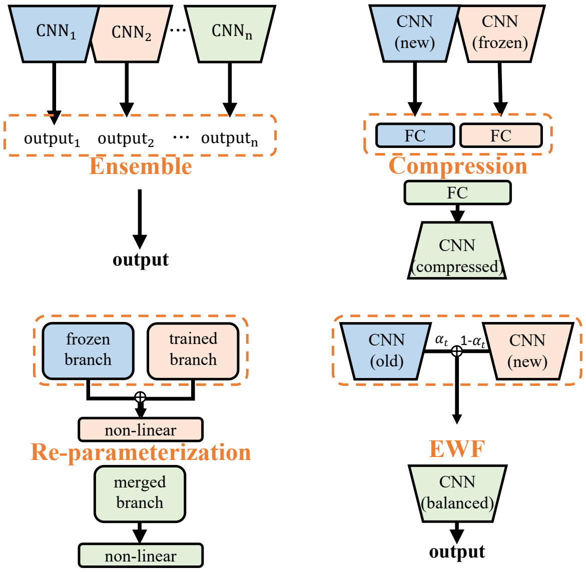


Figure 1. Illustration of different fusion strategies for incremental learning. *Ensemble* methods utilize multiple models to accumulate more knowledge. *Compression* methods reduce the model size and distill the knowledge into a small network. While *Re-parameterization* methods use equivalent operations for model fusion. Our Endpoints Weight Fusion (*EWF*) proposes model addition with a dynamic factor (α_t) with no further training.

To alleviate the problem of catastrophic forgetting without extra labeling or training cost, class incremental semantic segmentation (CISS) [3, 16, 50] aims at optimizing the trade-off between maintaining discrimination for old classes and learning knowledge of new classes. Most works [3, 16, 17, 37] designed regularization methods to maintain a balance between memorizing old knowledge and learning new one. We observe that existing works can still suffer from catastrophic forgetting, resulting in a significant

[*] The first two authors contribute equally.
[†] Corresponding author (xialei@nankai.edu.cn)

performance drop in old classes. In the scenario of CISS, not only the previous data is not accessible due to privacy issues or data storage limitations, but regions of old classes in the newly added dataset are labeled as background, which further exacerbates the model over-fitting.

Besides, training a new model from the old one and fusing them to obtain the final model is a common strategy in continual learning. As shown in Fig. 1, we roughly divide them into four categories with two stages of model expansion and fusion. Some methods [27, 33, 42, 47] propose to expand the model in incremental steps and *ensemble* the old and new outputs, which have large memory and inference costs. While some works apply *compression* [46, 47] to compress the old and new model to a unified model with fewer parameters. Nevertheless, these require further training on only new data, which can lead to a bias toward new data. Subsequently, some works [50, 53] explore knowledge decoupling and perform linear parameter fusion with *re-parameterization*. However, this is an intra-module fusion strategy, which is restricted to certain operations. As the last category, we propose Endpoints Weight Fusion (EWF) in the form of parameter addition between the old and new model with a dynamic factor, which requires no further training and re-parameterization, and maintains a constant model size as more tasks are encountered.

In this work, we adapt weight fusion to CISS and propose the EWF strategy, which aims at utilizing weight fusion to find a new balance between old and new knowledge. During incremental training, we choose a starting point and an ending point model of the current task training trajectory. The starting point represents the old knowledge, while the ending point represents the new knowledge. After learning the current task, a dynamic weight fusion is proposed for efficient knowledge integration. We aggregate them by taking the weighted average of the corresponding parameters of the two models. Nevertheless, the training procedure without restraints on the model would increase the parameter distance between the start and end points, limiting the performance improvement brought by the EWF strategy. To overcome this shortcoming, we further enhance the EWF strategy with a knowledge distillation scheme [16, 17, 50], which can largely increase the similarity of the models at the two points and boost the efficiency of EWF.

To summarize, the main contributions of this paper are:

- We propose an Endpoints Weight Fusion strategy, which has no cost of further training and keeps the model size the same. It can effectively find a new balance between old and new categories and alleviate catastrophic forgetting.

- Our method can be easily integrated with several state-of-the-art methods. In several *CISS* scenarios of long sequences, it can boost the baseline performance by more than 20%.

- We conduct experiments on various *CISS* scenarios, which demonstrate that our method achieves the state-of-the-art performance on both PASCAL VOC and ADE20K.

2. Related Work

Class Incremental Learning. Class incremental learning mainly focuses on alleviating catastrophic forgetting while learning the discriminative information required for the newly coming classes. It is most commonly analyzed in image classification, and the techniques can be roughly divided into three categories [11]. Many works [43–45] focus on the structural properties of the model (*i.e., Structural-Based Method*). The idea is to freeze old models and expand the architecture space to learn new knowledge, which normally results in a growing capacity and memory size of the model. Another way is to regularize models during incremental learning (*i.e., Regularization-Based Method*) [7, 8, 12, 17], strengthening memory via constraints (*e.g.,* knowledge distillation [39, 41] or gradient penalty [26, 29]). These methods bring the negligible cost to the learning process, but they allow for less freedom for parameter updates. Some other methods [1, 2, 28] propose to review knowledge through rehearsal (*i.e., Rehearsal-Based Method*). They store old data and mix it with new data to re-train the model [5, 21, 52].

Class Incremental Semantic Segmentation. Semantic segmentation [19] aims at assigning different categories to each single pixel, and has recently attracted attention for learning in a class incremental learning scenario [3, 16]. Nevertheless, data for semantic segmentation takes more space [24, 48] to be stored compared to the classification problem. Therefore, recent work mainly concentrates on utilizing distillation to transfer old knowledge to the new model without saving exemplars from old tasks.

MiB [4] proposes to model the potential classes to tackle semantic drift. PLOP [16] applies feature distillation to restrict representation ability. SDR [37] uses prototype matching to strengthen consistency in latent space. And RC-IL [50] analyzes the disadvantages of strip pooling and proposes average-pooling-based distillation to back up training. On the contrary, SSUL [6] does not apply distillation and proposes to fix the feature extractor instead of updating its parameters. In addition, they introduce thousands of extra data to help generate pseudo labels. But, simply fixing the model definitely does damage to the balance between plasticity and stability, and it is not sustainable when facing huge amounts of newly coming data. On the other hand, only applying distillation can restrict performance, since it can just limit the representation of new data to be the same. Summarizing the above thoughts and

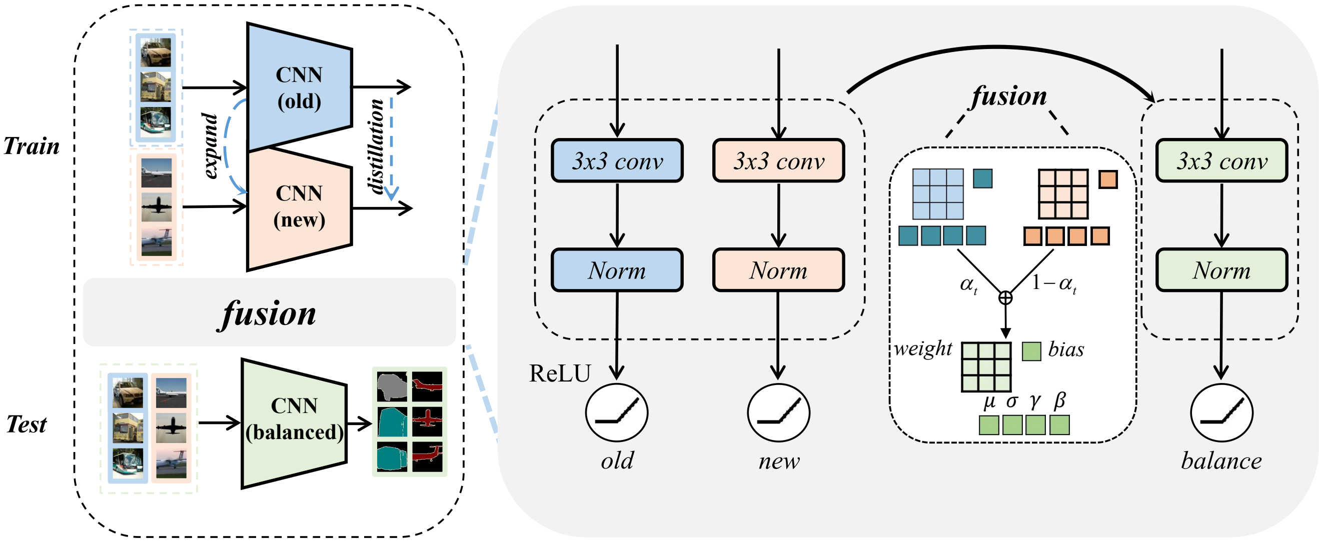


Figure 2. Illustration of our Endpoints Weight Fusion (EWF) framework. On the left side, we illustrate training and testing processes. Training is enhanced by knowledge distillation and fusion is applied after training to accumulate all seen knowledge. On the right side, given a 3×3 convolution layer and a normalization layer, they are fused by using a dynamic trade-off parameter α_t. Note that there is no computation cost for our fusion process.

inspired by RC-IL [50], we design a strategy different from the above, cooperating with distillation and maintaining the balance between the old and new knowledge with model weight fusion.

Weight Fusion Method. Weight Fusion is widely used in neural network training to improve performance. It can be applied with both linear and nonlinear operations. In linear mode, RCM [27] explores the additivity of convolution and applies it on multi-task learning. ACNet [14] and RepVGG [15] first propose structural re-parameterization, which is used to merge multi-branch Convolution-Batch Normalization serial sequences to a plain Convolution layer. RC-IL ingeniously exploits this operation to establish a representation compensation mechanism in continual learning. In nonlinear mode, weight averaging is commonly used to bring closer connections between different models. Famous methods like BYOL [20] use EMA [38] to improve the knowledge transfer effect or model robustness. From this point, we design a strategy to reach a new balance, which has no effect on the current training process and fully releases the learning ability of the model.

3. Method

3.1. Preliminaries

We consider a multi-step training process where T tasks are sequentially learned by model f_θ in a fully supervised scenario for semantic segmentation, where f is parameterized by parameter θ. Here f_{θ_t} denotes the model at task t. Each task contains a dataset $\mathcal{D}_t = \{x_i, y_i\}$, where x_i denotes the data and y_i denotes the corresponding label. The training label space of the task T is denoted as $C_t \bigcup \{c_b\}$, where C_t includes all classes that appear in this task, and c_b represents the background. Since $C_i \cap C_j = \emptyset$, the objective of different tasks are not the same, easily leading to catastrophic forgetting. To save annotation costs, only the categories that need to be learned at this stage will be labeled. Thus, c_b contains not only the real background, but also the classes appearing in past and future tasks. This complicates the training of the model f_θ since the background label c_b can refer to different classes at different tasks. This exacerbates the severity of forgetting.

3.2. Endpoints Weight Fusion (EWF)

As illustrated in Fig. 1, the existing model expansion-fusion methods have their own drawbacks for enhancing the model's memory of old knowledge. Thus, to better retain the old knowledge while boosting the learning ability of the model, we introduce our Endpoints Weight Fusion strategy. Considering the training process at step t, we choose a starting point model and an ending point model. The final model of the previous step (θ_{old}) is regarded as the best container of old knowledge. While the model after training on the current task (θ_{new}) contains the discriminative information for new classes. Additionally, we introduce another parameter α for the endpoints weight fusion, aiming at fusing two sets of parameters in a certain ratio. This ratio can be seen as a balance factor. Therefore, the operation of endpoints weight fusion can be written as:

$$\theta_{balanced} = \alpha_t \theta_{new} + (1 - \alpha_t)\theta_{old} \tag{1}$$

where the $\theta_{balanced}$ denotes the final model of this task, and the $\theta_{old}, \theta_{new}$ are defined as above. Furthermore, the number of incremental categories (denoted as N_{new}) and original categories (denoted as N_{old}) to a certain extent represent the degree of emphasis on the θ_{new} and θ_{old}. For the selection of α_t, we need to consider N_{new} and N_{old} in the current learning step. We decide to replace the constant ratio with an equation related to N_{new} and N_{old}. In detail, this formula can be expressed as:

$$\alpha_t = \sqrt{\frac{N_{new}}{N_{new} + N_{old}}} \tag{2}$$

It can be used across different tasks and scenarios. It can adapt to each task in CISS with a dynamic factor. We illustrate the main idea of our method in Fig. 2.

Knowledge distillation enhanced for EWF. In practice, training without any constraint will significantly increase the distance between different models and break the similarity of model representations. That means it deteriorates the forgetting of the model, which further undermines the model's ability to discriminate new classes. Moreover, guaranteeing a low error linear path between two distant models is an overly strong assumption. Thus, choosing a model trained without constraints as an ending point is potentially harmful, and we, therefore, introduce knowledge distillation to enhance the compatibility of the models of our Endpoints Weight Fusion method.

Knowledge distillation is a commonly used technique to prevent models from forgetting. As stated above, we utilize distillation to back up our strategy, limiting the distance between two endpoints and forcing them to be similar. In general, distillation used in continual learning is mainly divided into two categories, *i.e.,* feature-based distillation and logit-based distillation. They can be represented as:

$$\begin{aligned} L_{FD} &= \frac{1}{|D|}\sum_{(x_i,y_i)\sim D} ||\Psi_{old}(x_i) - \Psi_{new}(x_i)||^2 \\ L_{LD} &= \frac{1}{|D|}\sum_{(x_i,y_i)\sim D} KL(\Phi_{old}(\Psi_{old}(x_i)), \Phi_{new}(\Psi_{new}(x_i))) \end{aligned} \tag{3}$$

Ψ_{old}/Φ_{old} and Ψ_{new}/Φ_{new} denote old and new feature extractor/classifier respectively, and D is the corresponding dataset of the incremental learning step. In CISS, two popular distillation losses (*i.e.,* UNKD, POD) are proposed by [3, 16], respectively. The former is a logit-based distillation and the latter is a feature-based distillation. They can be easily integrated into our method.

Discussion on EMA v.s. EWF. An update strategy similar to our method is using EMA [38] to update the model. Here we will discuss the difference between our method EWF and EMA, and the advantages of EWF. The EMA strategy maintains a moving average model during training, and uses this model to replace the final model for inference. The moving average model can be represented as:

$$v^i = \beta v^{i-1} + (1-\beta)\theta^i \tag{4}$$

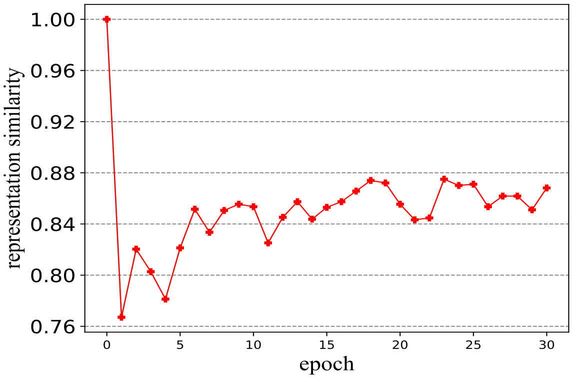


Figure 3. Representation similarity between old model θ^1 before training and being trained new model θ^i. The similarity measure we use is cosine similarity. We extract the intermediate results of the feature extractors of the old and new networks and calculate the average of the cosine similarity between them.

where the v^i denotes the moving average of the first i models, and θ^i denotes the model after ith iteration. β is the moving average parameter, which is usually set as 0.9/0.99. It is easy to see that the EMA is an inter-iteration operation, which is easier to realize due to the low distance between models. Since we use SGD as an optimizer, following Eq. 5, the results of EMA and EWF can be represented as v^n and $\theta_{balanced}$ in Eq. 6. Note that the learning rate does not affect the result of the analysis, so we set the learning rate to 1. The more detailed derivations are in the Appendix.

$$\begin{aligned} \theta^n &= \theta^{n-1} - \nabla_L(\theta^{n-1}) \\ &= \theta^1 - \sum_{k=1}^{n} \nabla_L(\theta^k) \end{aligned} \tag{5}$$

$$\begin{aligned} v^n &= \theta^1 - \sum_{k=1}^{n-1}(1-\beta^{n-k})\nabla_L(\theta^k) \\ \theta_{balanced} &= \theta^1 - \alpha\sum_{k=1}^{n-1}\nabla_L(\theta^k) \end{aligned} \tag{6}$$

Since the β is usually set to 0.9/0.99 and the α is set following Eq. 2, α is always smaller than β. According to Eq. 6, EMA gives more weight to the gradient of early time, and then gradually attenuates the influence of gradients in subsequent iterations. On the contrary, EWF gives the gradient of different parts a relatively uniform weight. Besides, to better observe the stability of incremental learning step, we calculate the representations similarity between θ^1 and θ^i. According to our observation in an incremental step, as shown in Fig. 3, the similarity of representations between θ^1 and θ^i first decreases and then increases, and gradually stabilizes in the subsequent stages. This reveals that in order

Algorithm 1 Pseudo Code for EWF in incremental steps

```
Require: $f, \theta_0, T, D_T$ and learning rate $\gamma$
  $t \leftarrow 1$
  while $t \leq T$ do
    Initialize $N_{new}, N_{old}$
    $\theta^1 \leftarrow \theta_{t-1}$
    $\alpha_t \leftarrow \sqrt{\frac{N_{new}}{N_{new}+N_{old}}}$
    $i \leftarrow 1$
    while not converged do
      Sample mini-batch $\{x_i, y_i\} \sim D$
      $\theta^{i+1} \leftarrow \theta^i - \gamma \nabla_{L_{CE}+L_{KD}}(f_{\theta^i})$
      $i \leftarrow i + 1$
    end while
    $\theta_{old} \leftarrow \theta^1, \theta_{new} \leftarrow \theta^i$
    $\theta_{balanced} \leftarrow \alpha_t \theta_{new} + (1 - \alpha_t)\theta_{old}$
    $\theta_t \leftarrow \theta_{balanced}$
    $t \leftarrow t + 1$
  end while
```

to learn new knowledge, the representation will first collapse and then recover under the action of the distillation and cross-entropy losses. Then EMA concentrates more of the gradient on the early *collapsed* process, while our method pays more attention to the useful gradient information after the *recovery*. From this perspective, our method is theoretically better than EMA in CISS.

3.3. Overall Framework

As stated above, to remember knowledge from old tasks, we borrow knowledge distillation to strengthen the model's memory. To learn the discrimination of new classes, we use the Cross-Entropy loss to optimize the model. In general, the objective is given by:

$$\min_{\theta_t} \mathcal{L}_{CE}(\theta_t) + \mathcal{L}_{KD}(\theta_t; \theta_{t-1}) \tag{7}$$

And the overall algorithm of EWF is shown in Alg. 1.

4. Experiments

We demonstrate experimental protocols, scenarios and training details. Furthermore, we evaluate our algorithm through quantitative and qualitative experiments.

4.1. Experimental setups

Protocols. In general, the training for CISS is divided into T steps, and each step denotes a task, where the labeled classes are disjoint in each of them. We adopt the *overlapped* setting as other works, in which the current training data may contain potential classes labeled as background in previous steps. The *overlapped* setting is more realistic, and therefore we only evaluate on this setup as previous methods [6, 16]. Following existing works [3, 16, 50], we conduct experiments on two widely used segmentation datasets, PASCAL VOC 2012 [18] and ADE20K [51]. The PASCAL VOC 2012 dataset [18] contains 10,582 training images and 1449 validation images with 20 object classes and the background class. The ADE20K dataset [51] is composed of 150 classes and contains 20, 210 training images, and 2000 validation images. Following previous works [3,16,50], $X - Y$ denotes different settings for CISS. In the $X - Y$ setting, the model can recognize X classes in the initial step, and then is supposed to learn Y newly added classes in each following step. At each step, only current task data is available for training. We perform experiments on PASCAL VOC 2012 [18] with four settings, 15-1, 10-1, 5-3 and 19-1. On ADE20K [51], we verify the effectiveness of our method on three settings, 100-5, 100-10, and 100-50.

Implementation Details. Following existing works [3, 16, 50], we apply Deeplab-v3 [9] as our segmentation model with ResNet-101 [23] as a backbone. We also use in-place activated batch normalization [40] in the backbone. In our experiments, we use some data augmentations, including horizontal flip and random crop. The ratio α_t for EWF is defined as Eq.2. Using SGD optimizer, we train the model for 30 (PASCAL VOC 2012) and 60 (ADE20K) epochs in each step with a batch size of 24. We set the initial learning rate as 0.01 for the first training step and 0.001 for the next continual learning steps. All experiments are conducted on 4 RTX 2080Ti GPUs. The learning rate is decayed with *poly* schedule. During training, we use 20% of the training set as validation, and report the mean Intersect over Union (mIoU) on the original validation set.

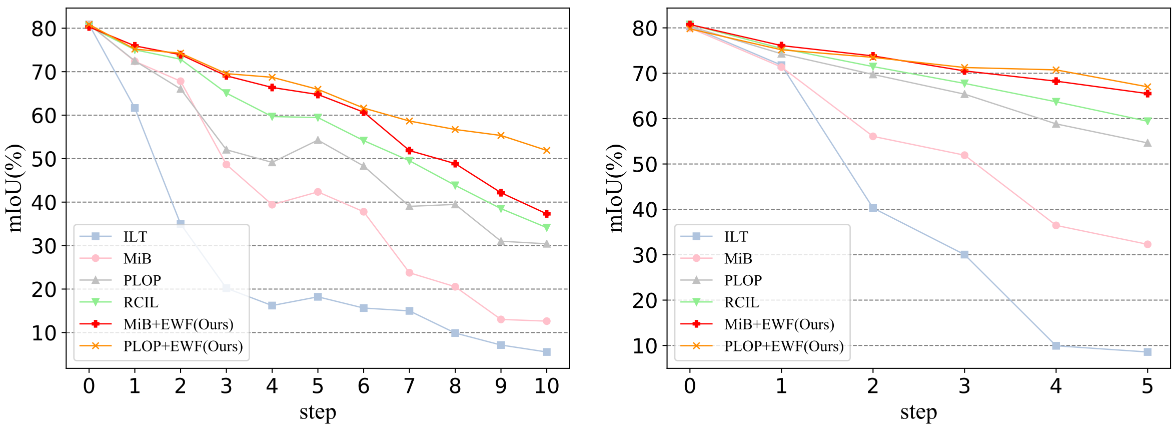
4.2. Comparison to competing methods

In this section, we apply our method to MiB [3] and PLOP [16]. Additionally, we also compare our method with LwF [31], ILT [36], SDR [37], and RCIL [50].

PASCAL VOC 2012. In this dataset, we use the same experimental settings as [3, 16, 50], we performed the experiments with the class incremental learning settings *15-1*, *10-1*, *5-3*, *19-1*. As shown in Table 1, we report the result of our experiments on the final task. From the results, we can observe that our method improves the results of both MiB and PLOP by a large margin, on the more challenging settings (*e.g., 15-1, 10-1, 5-3*. For instance, on the *15-1* setting, our algorithm obtains performance gains of 12.4% and 33.3% mIoU for PLOP and MIB, respectively. Furthermore, on the longest learning sequence, the *10-1* setting, our method obtains a large gain consistently, improving the performance of PLOP and MIB by 21.4% and 24.7%, respectively. In Table 1 we also report the performance of the old and new classes for different settings. Our method achieves significant performance gains for old classes. This demon-

Method	**15-1** (6 steps)			**10-1** (11 steps)			**5-3** (6 steps)			**19-1** (2 steps)		
	0-15	*16-20*	*all*	*0-10*	*11-20*	*all*	*0-5*	*6-20*	*all*	*0-19*	*20*	*all*
LwF [31] (TPAMI2017)	6.0	3.9	5.5	8.0	2.0	4.8	20.9	36.7	24.7	53.0	8.5	50.9
ILT [36] (ICCVW2019)	9.6	7.8	9.2	7.2	3.7	5.5	22.5	31.7	29.0	68.2	12.3	65.5
SDR [37] (CVPR2021)	47.3	14.7	39.5	32.4	17.1	25.1	-	-	-	69.1	32.6	67.4
RCIL [50] (CVPR2022)	70.6	23.7	59.4	55.4	15.1	34.3	63.1	34.6	42.8	77.0	31.5	74.7
MiB [3] (CVPR2020)	38.0	13.5	32.2	12.2	13.1	12.6	57.1	42.5	46.7	71.2	22.1	68.9
MiB+**EWF (Ours)**	78.0	25.5	65.5	56.0	16.7	37.3	69.0	45.0	**51.8**	77.8	12.2	**74.7**
PLOP [16] (CVPR2021)	65.1	21.1	54.6	44.0	15.5	30.5	25.7	30.0	28.7	75.4	37.3	73.5
PLOP+**EWF (Ours)**	77.7	32.7	**67.0**	71.5	30.3	**51.9**	61.7	42.2	47.7	77.9	6.7	74.5
Joint	79.8	72.6	78.2	79.8	72.6	78.2	78.2	78.0	78.2	76.9	77.6	77.4

Table 1. The mIoU(%) of the last step on the Pascal VOC 2012 dataset for different class-incremental segmentation scenarios.



(a) *10-1* scenario on PASCAL VOC

(b) *15-1* scenario on PASCAL VOC

Figure 4. The mIoU (%) at each step for the settings *10-1* (a) and *15-1* (b).

strates that our EWF strategy can enhance significantly the model's memory of old knowledge (by successfully countering forgetting). In the more challenging settings, *e.g.,* 15-1, 5-3, 10-1, our method also boosts the performance of new classes. This shows that EWF can achieve high plasticity on new steps, and that the proposed dynamically weighted combination of the network allows achieving a good trade-off between plasticity and stability. We also show dynamic performance changes during the continual learning process in Fig. 4. It is clear that with more learning steps, the gap between our method and the best baseline is growing, and that the curve of our method for different settings (*15-1* and *5-3*) is on the top throughout most of the learning trajectory.

Comparison with methods that introduce auxiliary data. It is worth noting that there are several methods [6, 34, 49] that introduce different forms of auxiliary data to assist continual semantic segmentation, helping the model build better pseudo-labels or enhance memory for old knowledge. RECALL [34] learns a generative model or retrieves from web-crawled data for replay, while SSUL [6] leverages salient object detectors (trained on MSRA-B dataset [32] with 5000 images) to generate saliency maps as auxiliary data. And ST-CIL [49] exploits unlabelled datasets with pseudo labels as ground truth. Even if our algorithm is designed to deal with extra-data-free scenarios, to further demonstrate the effectiveness of our method, we integrate our method to SSUL and compare it with the above methods in Table 2. Note that SSUL freezes the backbone completely and it is hard to directly apply our method on it. Therefore, we learned part of the backbone network parameters when we apply our method to SSUL. In detail, we set the second stage of the backbone in SSUL as learnable parameters for model fusion. SSUL obtains the best performance among these methods using auxiliary data, and our method applied to SSUL can further improve it by about 1% for all classes.

Visualization. As shown in Fig. 5, we compare our method based on MiB, and we show a sample from the base task (step 0) and a sample from step 2. From the top two rows, the person and bike classes are largely preserved with our method, but it forgets gradually for the baseline MiB. From the bottom two rows, it shows an example with both old classes and a new sheep class. Our method can keep the old knowledge better while incorporating new knowledge.

ADE20K. In order to further evaluate the effectiveness of

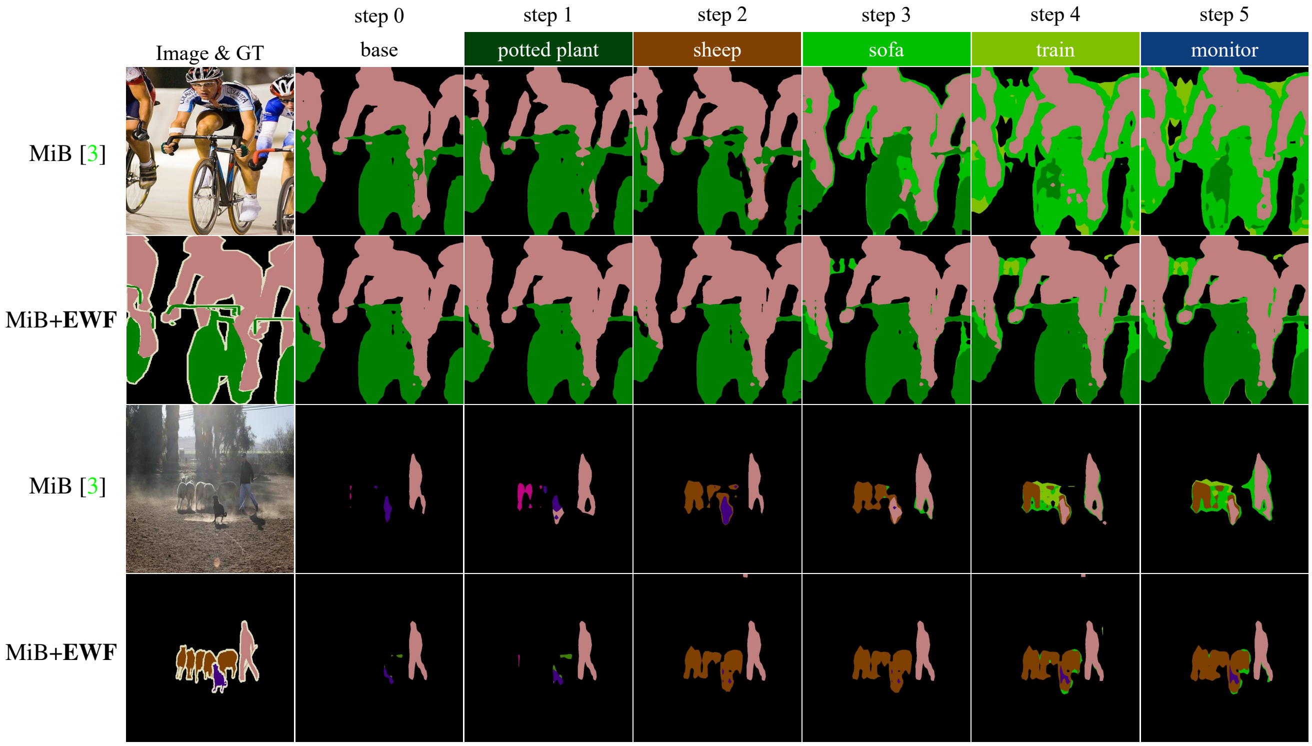


Figure 5. The qualitative comparison between different methods. All prediction results are from the last step of 15-1 overlapped setting. In the initial step, 15 classes are learned and 5 tasks are learned incrementally with class potted plant, sheep, sofa, train and monitor.

		15-1 (PASCAL VOC 2012)			**10-1** (PASCAL VOC 2012)		
Method	**Auxiliary Data**	*0-15*	*16-20*	*all*	*0-10*	*11-20*	*all*
RECALL [34] (ICCV2021)	GAN / Web-Data	65.7	47.8	62.7	59.5	46.7	54.8
ST-CIL [49] (TNNLS2022)	Unlabeled Data	71.4	40.0	63.6	-	-	-
SSUL [6] (NeurIPS2021)	Saliency Map	77.3	36.6	67.6	71.3	45.9	59.2
SSUL + EWF (**Ours**)	Saliency Map	77.9	38.9	**68.6**	72.4	47.4	**60.5**

Table 2. The mIoU(%) of the last step on the Pascal VOC 2012 *15-1* and *10-1* overlapped setting.

our method, we conduct experiments on ADE20K dataset. In Table 3, we show the experimental results with settings *100-50*, *100-10* and *100-5*. As shown in Table 3, our method reaches superior performance on this dataset. Especially on the most challenging settings *100-5* and *100-10*, our method achieves 6.2% and 3.0% improvement over MiB [3], respectively. It also surpasses the state-of-the-art method RC-IL by a large margin on *100-5* setting. This indicates that our EWF is effective on large-scale datasets as well.

4.3. Ablation Study

In this part, we demonstrate and analyze the effectiveness of weight fusion and its dynamic factor selection. We use MiB [3] for the ablation experiments.

Fusion Strategy. In Table 4, we demonstrate the performance of different fusion strategies. These experiments are conducted with the setting *15-1* on PASCAL VOC 2012. Most of these methods perform continual learning by fusing the information present in the previous step model. Among these, EMA [38] updates a stored set of parameters with moving average during training, and uses it for inference. Model ensemble fuses the previous model's prediction and new prediction in an average way during inference. And None denotes the model trained only with Distillation and Cross-Entropy (*i.e.,* MiB [3]). Compared with no fusion method, EMA and model ensemble have 5.1% and 5.0% performance improvement, respectively. While Our EWF has a 33.3% improvement on the basis of a simple fusion strategy, which again verifies the discussions in Sec. 3.2.

Fusion Factor Selection. In this part, we conduct experiments on the fusion factor selection. To prove the advantages of our parameter selection strategy, we use some fixed values as balance parameters in the fusion process, and we compare the difference between our method and theirs to evaluate its advantage. In detail, we choose three scenarios

	100-50 (2 steps)			**100-10** (6 steps)							**100-5** (3 steps)		
Method	*1-100*	*101-150*	*all*	*1-100*	*101-110*	*111-120*	*121-130*	*131-140*	*141-150*	*all*	*1-100*	*101-150*	*all*
ILT [36] (ICCVW2019)	18.3	14.8	17.0	0.1	0.0	0.1	0.9	4.1	9.3	1.1	0.1	1.3	0.5
PLOP [16] (CVPR2021)	41.9	14.9	32.9	40.6	15.2	16.9	18.7	11.9	7.9	31.6	39.1	7.8	28.7
RC-IL [50] (CVPR2022)	42.3	18.8	34.5	39.3	14.6	26.3	23.2	12.1	11.8	32.1	38.5	11.5	29.6
MiB [3] (CVPR2020)	40.7	17.7	32.8	38.3	12.6	10.6	8.7	9.5	15.1	29.2	36.0	5.6	25.9
MiB+**EWF(Ours)**	41.2	21.3	**34.6**	41.5	12.8	22.5	23.2	14.4	8.8	**33.2**	41.4	13.4	**32.1**
Joint	44.3	28.2	38.9	44.3	26.1	42.8	26.7	28.1	17.3	38.9	44.3	28.2	38.9

Table 3. The mIoU(%) of the last step on the ADE20K dataset for different overlapped continual learning scenarios.

Fusion strategy	$step_1$	$step_2$	$step_3$	$step_4$	$step_5$
None [3]	71.4	56.1	51.9	36.5	32.2
EMA [38]	74.0	59.6	59.8	40.9	37.3
model ensemble [13]	74.1	60.1	60.3	41.5	37.2
EWF (Ours)	76.1	73.8	70.5	68.3	65.6

Table 4. Ablation study of fusion strategies. All performances are reported on PASCAL VOC 2012 *15-1* setting.

Parameter Selection	Ours	0.2	0.4	0.6	0.8
15-1	65.6	65.6	63.7	60.1	53.3
10-1	37.3	39.5	31.8	22.7	14.3
5-3	51.8	38.0	51.2	56.1	52.9
Average	**51.6**	47.7	48.9	46.3	40.2

Table 5. Comparison between our dynamic parameter fusion strategy and fixed balance factors.

to conduct our experiments (*i.e., 15-1, 10-1, 5-3*), and we calculate the average mIoU of different strategies for three scenarios to measure the final performance. Since the balance factor $\alpha \in [0, 1]$, we take 0.2, 0.4, 0.6, and 0.8 as the fixed values to compare to our parameter selection strategy. As shown in Table 5, for the *15-1* setting, our method reaches the highest performance compared with other fixed parameters. In addition, although our method is slightly below the highest performance for fixed parameters on other settings, the parameters for the highest performance on different settings vary widely. It means that choosing a fixed parameter for all scenarios is unrealistic and harmful to the algorithm. Importantly, our method reaches the highest average performance among all other fixed values, which indicates that our strategy can be easily applied to different settings without tuning the hyper-parameters manually.

Robustness of Class Order. In the scenario of class incremental semantic segmentation, the order of classes encountered by the model is significant in measuring the effectiveness of an algorithm. Thus, to verify the effectiveness of our algorithm and the robustness to different class orders, we conduct experiments on five different class orders to calculate the average performance and their standard deviation. As shown in Fig. 6, our method significantly improves the performance while also significantly enhancing the robustness to different class orders.

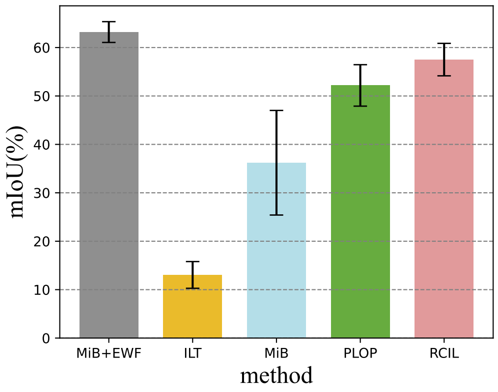


Figure 6. Illustration on robustness with respect to different orders.

5. Conclusions

In this work, we address the class incremental semantic segmentation (CISS) problem with a simple yet effective endpoints weight fusion method. It is enhanced by the existing distillation-based methods and easily integrated with them. A dynamic parameter fusion strategy is proven to be flexible for different settings and it avoids the further tuning of hyper-parameters. Interestingly, we discuss the relationship between our method and a popular weight fusion method EMA, which reveals why our method is more effective in incremental learning. The experimental results demonstrate that our method can obtain a significant gain compared to the baselines and achieve superior performance. In future work, we will investigate further the underlining reasons why our simple EWF strategy works so well in CISS. Moreover, we are planning to evaluate our incremental learning strategies for other application domains.

Acknowledgment. This work is funded by NSFC (NO. 62225604 and 62206135), Fundamental Research Funds for the Central Universities (Nankai University, NO. 63223050), and Bytedance Limited. We acknowledge the Spanish Government funding for projects PID2019-104174GB-I00, TED2021-132513B-I00.

References

[1] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, 2021. 2

[2] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *ICCV*, pages 583–592, 2019. 2

[3] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *CVPR*, pages 9233–9242, 2020. 1, 2, 4, 5, 6, 7, 8

[4] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *CVPR*, pages 9233–9242, 2020. 2

[5] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *ICCV*, pages 9516–9525, 2021. 2

[6] Sungmin Cha, YoungJoon Yoo, Taesup Moon, et al. Ssul: Semantic segmentation with unknown label for exemplar-based class-incremental learning. *NeurIPS*, 34, 2021. 2, 5, 6, 7

[7] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, 2018. 2

[8] Arslan Chaudhry, Albert Gordo, Puneet Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. In *AAAI*, volume 35, pages 6993–7001, 2021. 2

[9] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 40(4):834–848, 2017. 5

[10] Yi-Hsin Chen, Wei-Yu Chen, Yu-Ting Chen, Bo-Cheng Tsai, Yu-Chiang Frank Wang, and Min Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *ICCV*, pages 1992–2001, 2017. 1

[11] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE TPAMI*, 2021. 2

[12] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *CVPR*, 2019. 2

[13] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000. 8

[14] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks. In *ICCV*, October 2019. 3

[15] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *CVPR*, 2021. 3

[16] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. Plop: Learning without forgetting for continual semantic segmentation. In *CVPR*, 2021. 1, 2, 4, 5, 6, 8

[17] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, volume 12365, pages 86–102, 2020. 1, 2

[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 5

[19] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and Jose Garcia-Rodriguez. A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*, 2017. 2

[20] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *NeurIPS*, 33:21271–21284, 2020. 3

[21] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *ECCV*, pages 466–483, 2020. 2

[22] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *ICCV*, pages 4918–4927, 2019. 1

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[24] Zilong Huang, Wentian Hao, Xinggang Wang, Mingyuan Tao, Jianqiang Huang, Wenyu Liu, and Xian-Sheng Hua. Half-real half-fake distillation for class-incremental semantic segmentation. *arXiv preprint arXiv:2104.00875*, 2021. 2

[25] Christian Häne, Christopher Zach, Andrea Cohen, and Marc Pollefeys. Dense semantic 3d reconstruction. *IEEE TPAMI*, 39(9):1730–1743, 2017. 1

[26] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *ECCV*, pages 699–715, 2020. 2

[27] Menelaos Kanakis, David Bruggemann, Suman Saha, Stamatios Georgoulis, Anton Obukhov, and Luc Van Gool. Reparameterizing convolutions for incremental multi-task learning without task interference. In *ECCV*, pages 689–707, 2020. 2, 3

[28] Chris Dongjoo Kim, Jinseo Jeong, Sangwoo Moon, and Gunhee Kim. Continual learning on noisy data streams via self-purified replay. In *ICCV*, pages 537–547, 2021. 2

[29] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 2

[30] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *CVPR*, pages 2661–2671, 2019. 1

[31] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, 2017. 5, 6

[32] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *IEEE TPAMI*, 33(2):353–367, 2010. 6

[33] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *CVPR*, 2021. 2

[34] Andrea Maracani, Umberto Michieli, Marco Toldo, and Pietro Zanuttigh. Recall: Replay-based continual learning in semantic segmentation. In *ICCV*, pages 7026–7035, 2021. 6, 7

[35] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *PsychologLearniny of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 1

[36] Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. In *ICCVW*, 2019. 5, 6, 8

[37] Umberto Michieli and Pietro Zanuttigh. Continual semantic segmentation via repulsion-attraction of sparse and disentangled latent representations. In *CVPR*, 2021. 1, 2, 5, 6

[38] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992. 3, 4, 7, 8

[39] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 2

[40] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. In-place activated batchnorm for memory-optimized training of dnns. In *CVPR*, 2018. 5

[41] Christian Simon, Piotr Koniusz, and Mehrtash Harandi. On learning the geodesic path for incremental learning. In *CVPR*, 2021. 2

[42] Pravendra Singh, Pratik Mazumder, Piyush Rai, and Vinay P Namboodiri. Rectification-based knowledge retention for continual learning. In *CVPR*, pages 15282–15291, 2021. 2

[43] Pravendra Singh, Vinay Kumar Verma, Pratik Mazumder, Lawrence Carin, and Piyush Rai. Calibrating cnns for lifelong learning. In *NeurIPS*, volume 33, 2020. 2

[44] James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In *ICCV*, 2021. 2

[45] Vinay Kumar Verma, Kevin J Liang, Nikhil Mehta, Piyush Rai, and Lawrence Carin. Efficient feature transformations for discriminative and generative continual learning. In *CVPR*, 2021. 2

[46] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. *arXiv preprint arXiv:2204.04662*, 2022. 2

[47] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *CVPR*, 2021. 2

[48] Shipeng Yan, Jiale Zhou, Jiangwei Xie, Songyang Zhang, and Xuming He. An em framework for online incremental learning of semantic segmentation. In *ACM MM*, pages 3052–3060, 2021. 2

[49] Lu Yu, Xialei Liu, and Joost Van de Weijer. Self-training for class-incremental semantic segmentation. *TNNLS*, 2022. 6, 7

[50] Chang-Bin Zhang, Jia-Wen Xiao, Xialei Liu, Ying-Cong Chen, and Ming-Ming Cheng. Representation compensation networks for continual semantic segmentation. In *CVPR*, 2022. 1, 2, 3, 5, 6, 8

[51] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 5

[52] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *CVPR*, pages 5871–5880, 2021. 2

[53] Kai Zhu, Wei Zhai, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Self-sustaining representation expansion for non-exemplar class-incremental learning. In *CVPR*, pages 9296–9305, 2022. 2