Class Similarity Weighted Knowledge Distillation for Continual Semantic Segmentation

Minh Hieu Phan[1], The-Anh Ta[2], Son Lam Phung[1,3], Long Tran-Thanh[4], Abdesselam Bouzerdoum[1,5],
[1]University of Wollongong, [2]FPT Software, AIC, [3]VinAI Research,
[4]University of Warwick, [5]Hamad Bin Khalifa University,
vmhp806@uowmail.edu.au, anhtt71@fsoft.com.vn,
{phung, a.bouzerdoum}@uow.edu.au, long.tran-thanh@warwick.ac.uk

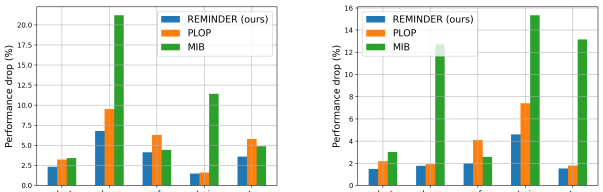
Abstract

Deep learning models are known to suffer from the problem of catastrophic forgetting when they incrementally learn new classes. Continual learning for semantic segmentation (CSS) is an emerging field in computer vision. We identify a problem in CSS: A model tends to be confused between old and new classes that are visually similar, which makes it forget the old ones. To address this gap, we propose REMINDER - a new CSS framework and a novel class similarity knowledge distillation (CSW-KD) method. Our CSW-KD method distills the knowledge of a previous model on old classes that are similar to the new one. This provides two main benefits: (i) selectively revising old classes that are more likely to be forgotten, and (ii) better learning new classes by relating them with the previously seen classes. Extensive experiments on Pascal-VOC 2012 and ADE20k datasets show that our approach outperforms state-of-the-art methods on standard CSS settings by up to 7.07% *and* 8.49%*, respectively.*

1. Introduction

Semantic segmentation, which aims to assign each pixel of an image to its semantic class, is a fundamental task in computer vision. Segmentation models are critical to many real-world applications, such as self-driving cars [1, 17] and medical image diagnostics [14, 40]. In most practical cases, the model needs to continuously learn new data and adapt to the changes in the operating environment. However, continually learning new classes leads to catastrophic forgetting of old knowledge [12, 34]. In other words, the performance of newly retrained models degrades significantly on old tasks.

Research on continual learning for semantic segmentation (CSS) only emerged recently in medical imaging [27, 28] and general scene understanding [2, 9]. Besides forgetting, CSS also faces the *background shift* problem, where object classes from previous steps are shifted to the background at the current step [2].



(a) Animal group (b) Vehicle group

Figure 1. Performance drop (degree of forgetting) on old classes in the (a) *animal* and (b) *vehicle* group as a model learns a new class. Our method forgets less when learning a new similar class.

There are two main problems causing catastrophic forgetting in continual learning (CL). First, the model has a strong bias toward new classes [36]. In other words, objects of old classes are mispredicted as new ones. Second, the model tends to forget old classes that are visually similar to newly added classes. To investigate this problem, we divide classes in the Pascal-VOC 2012 dataset into two groups: animal and vehicle, and evaluate the degree of forgetting in each group. Fig. 1 shows the performance drop[1] in each group as the model learns a new class. The performance on the animal group drops the most when the model learns *sheep*. Similarly, the result on the vehicle group reduces the most when it learns *train*.

Recent CSS methods [2, 9, 23, 24] distill the knowledge of a previous model on old classes to a current model. Knowledge distillation prevents the model from diverging from what it previously learned. This continual learning paradigm gains high research interests because of its computational efficiency. They do not require storing exemplars of old classes to re-learn old knowledge. Despite the recent

[1]The performance drop measures how much mIoU a model drops in percentage point(%) when it learns a new class.

success, modern distillation based methods [2,23,24] distill the knowledge on all old classes equally even though some are more likely to be forgotten than others. They may put less emphasis on revising the affected old knowledge. This overlooking makes the model more vulnerable to forgetting visually similar old classes.

To resolve the current research gap, this paper proposes a novel class similarity weighted knowledge distillation (CSW-KD) method. Our CSW-KD emphasizes revising the knowledge of old classes that are likely to be forgotten, i.e., the classes that are similar to a new one. In particular, when learning a new class, the proposed method computes its similarity to the old ones. It then reweighs the predictions of a previous model on old classes based on their similarity scores. The class similarity weighted knowledge is distilled to the current model.

The proposed approach has three benefits. *First*, our method is more resilient to forgetting when learning new visually similar classes (as shown in Fig. 1). The model identifies the group of old classes that is more likely to be forgotten, i.e., the group to which a new class belongs. It then selectively reinforces the knowledge of this group. *Second*, our method better learns new tasks. Via CSW-KD, we enforce the model to capture the similarity between classes. Thus, it can relate the new with the previously learned knowledge. The model then transfers what it previously learned to facilitate the learning of new classes. *Third*, the prior knowledge about class similarity enables the model to learn an underlying *class hierarchy*. Using this learned hierarchy, the model can identify groups of old knowledge that are being affected.

We introduce REMINDER - a CSS framework that consists of two components. First, the class similarity weighted knowledge distillation (CSW-KD) transfers the reweighted outputs of an old model based on their similarity to the new class. Second, a feature knowledge distillation (FKD) module distills the features of the previous model to encourage feature reuse among different tasks.

Our main contributions can be summarized as follows.

- We propose to use semantic similarity between classes as a prior for continual learning. To the best of our knowledge, this is the first work that explores hierarchical learning to reduce catastrophic forgetting in CL.
- We propose a novel CSW-KD method that leverages class similarity to reduce the forgetting of similar old classes (rigidity) and promote learning of new classes (plasticity). We then propose REMINDER - a CSS framework that uses CSW-KD to remind the model of old knowledge based on the similarity between new and old classes.
- We show that our method achieves a better rigidity-plasticity trade-off than strong baselines via extensive experiments. REMINDER outperforms state-of-the-art methods on Pascal-VOC 2012 and ADE20k datasets by up to 7.07% and 8.49%.

2. Related Work

Continual learning. To reduce forgetting, popular methods in continual learning can be categorized into four main approaches. First, regularization techniques aim to apply penalty constraints on networks' weights to prevent catastrophic forgetting [4,7]. Second, replay-based methods propose to store a portion of data from old classes or generate training data from previous tasks [6, 33]. Then the model is trained on a mixture of new and old data. Third, dynamic architectures either grow new branches for new tasks or rearrange subnetworks for specific tasks [19,37]. Fourth, parameter isolation approaches train each task on its own different subset of weights to preserve model performance on old tasks [20,32].

A recent neuroscience study investigates how new knowledge is integrated into a neocortex-like network [22]. Their experiments show that the neural network implicitly learns a hierarchy. When learning new classes, the model projects them onto a known branch or creates a new branch in a hierarchy. Notably, replaying old items within the same branch as the new item results in a faster integration. Inspired by this study, our method learns a class hierarchy and selectively revises old items similar to new ones. Our selective knowledge revision improves the learning of new classes and reduces the forgetting of old classes.

Continual semantic segmentation. The common framework of continual learning for general semantic segmentation tasks was first proposed in [23] which uses distillation losses from output and feature spaces of a model from previous tasks to train on new tasks. Besides catastrophic forgetting, CSS faces the problem of background shift which was first pointed out in [2].

Recent approaches [2, 9, 24] adopt knowledge distillation techniques for CSS. An unbiased knowledge distillation (UNKD) method proposed in MiB [2] allows the old model to predict background pixels as one of a new class in the current task. Local pooled outputs distillation (local POD) is a recent state-of-the-art proposed in PLOP [9]. Local POD distills both long-range and short-range spatial relations across training steps to preserve multi-scale information for CSS. Sparse and disentangled representations (SDR) is a recent method that applies prototype matching and contrastive learning to improve feature robustness for CSS [24].

Contemporary works [16,21,39] have developed several replay-based methods for CSS. Half-real half-fake distillation proposes to generate synthetic images and add them to training data of new tasks to remind models about old classes [16]. RECALL uses generative adversarial networks

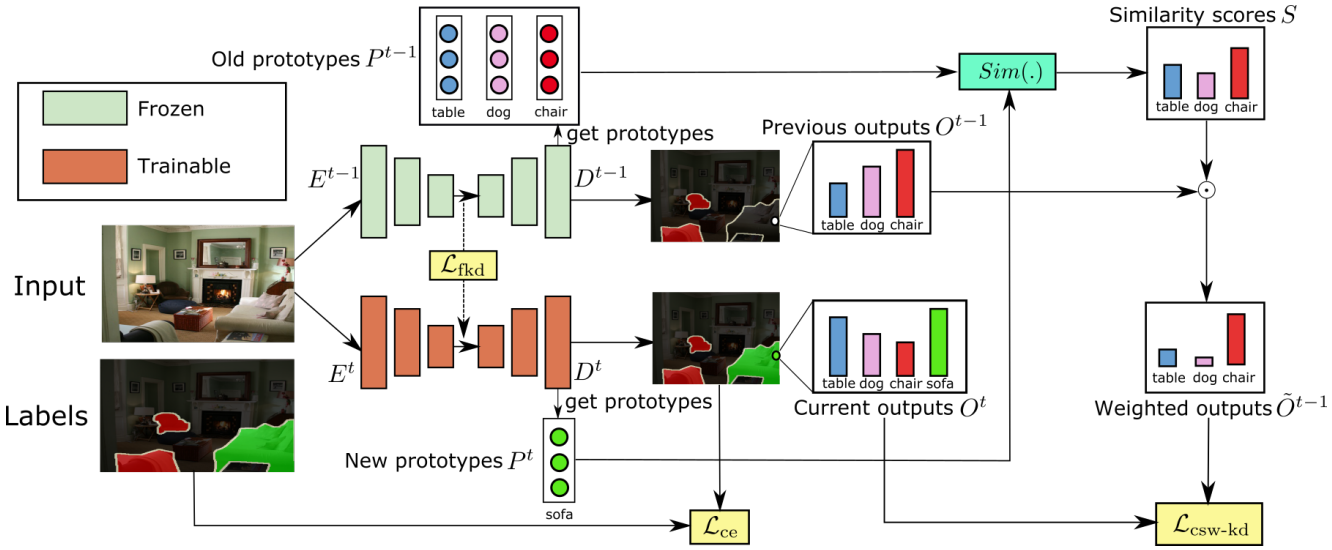


Figure 2. **Overview of REMINDER**. The model is trained via three loss objectives (in yellow): (i) a cross-entropy loss from labels $\mathcal{L}_{\text{ce}}$, (ii) an feature knowledge distillation (FKD) loss $\mathcal{L}_{\text{fkd}}$, and (iii) a class similarity weighted knowledge distillation (CSW-KD) loss $\mathcal{L}_{\text{csw-kd}}$. When a model learns a new class (i.e., sofa), our CSW-KD weighs the predictions score on old classes based on their similarity score S to the sofa. It then distills the reweighted outputs $\tilde{O}^{t-1}$ to the current model via CSW-KD loss $\mathcal{L}_{\text{csw-kd}}$.

and web-crawled data to regenerate new samples from old classes for training new tasks [21]. Another approach [39] proposes an expectation-maximization framework for CSS, which combines relabeling and replay-based approaches.

Prototype-based approach. Prototype-based approaches were initially proposed for few-shot learning [8, 35] and domain adaptation [6, 29, 38]. They extract prototypes of classes to distill general knowledge of each class or encourage orthogonality between classes. Recent CSS methods such as SDR [24] and PIFS [3] use prototypes to regularize features for reducing the forgetting of old classes. They enforce features to stay close to the corresponding prototypes in the current task. In contrast, our method compares prototypes across different tasks for identifying the classes that are likely to be forgotten.

3. Method

3.1. Problem definition and background

Problem definition. Continual semantic segmentation aims to train a segmentation model in T steps without forgetting. In step t, we are given a dataset $\mathcal{D}^t$ which comprises a set of pairs (X^t, Y^t), where X^t is an image of size $H \times W$ and Y^t is the ground-truth segmentation map. Here, Y^t only consists of labels in current classes $\mathcal{C}^t$, while all other classes (i.e., old classes $\mathcal{C}^{1:t-1}$ or future classes $\mathcal{C}^{t+1:T}$) are assigned to the current background class c_b. In continual learning, the model at step t should be able to predict all classes $\mathcal{C}^{1:t}$ in the history.

A segmentation model f_{θ^t} consists of an encoder E^t for extracting features and a decoder D^t for producing the segmentation map. The encoder extracts low-level features via L layers $f_l^t(.)$, where $l \in \{1, \dots, L\}$. The decoder learns high-level features from the encoder's features and outputs the logit map $Z^t = D^t \circ E^t(X^t)$. A softmax function is applied on Z^t to give a segmentation map O^t.

Revisiting knowledge distillation loss. To avoid storing old data, distillation loss [15] is applied to transfer the knowledge of the old model f_{θ^t} to the new one. Each image has a set of pixels $\mathcal{I}$ with the cardinality $|\mathcal{I}| = HW$. The distillation loss is formulated as

$$\mathcal{L}_{\text{kd}} = -\frac{1}{HW} \sum_{i=1}^{HW} \sum_{c \in \mathcal{C}^{1:t-1}} O_{i,c}^{t-1} \log O_{i,c}^t, \qquad (1)$$

where $O_{i,c}^t$, given by f_{θ^t}, refers to the probability of class c in pixel i at step t. Here, the output $O_{i,c}^t$ is defined by re-normalizing the logit Z^t across all classes in the previous step $\mathcal{C}^{t-1}$:

$$O_{i,c}^t = \begin{cases} 0, & \text{if } c \in \mathcal{C}^t \setminus b \\ \frac{\exp Z_{i,c}^t}{\sum_{k \in \mathcal{C}^{1:t-1}} \exp Z_{i,c}^t} & \text{if } c \in \mathcal{C}^{1:t-1}. \end{cases} \qquad (2)$$

The distillation loss in Eq. 1 encourages model f_{θ^t} at step t to produce similar outputs to model $f_{\theta^{t-1}}$ at step $t-1$. This enforces the parameters of f_{θ^t} to stay close to the solution found by $f_{\theta^{t-1}}$ for labeling pixels of previous classes.

3.2. Proposed REMINDER framework

We first train the model f_{θ^0} to recognize pixels belonging to the initial classes $\mathcal{C}^0$ using the cross-entropy loss. The

model $f_{\theta^{t-1}}$ trained on the previous step is frozen. Its knowledge is used to regularize the current model f_{θ^t} at step t. Fig. 2 illustrates our proposed REMINDER framework. Our approach trains the segmentation model using three losses: i) the cross-entropy loss from labels, ii) the encoder knowledge distillation loss from features of encoder E^{t-1}, and iii) the class similarity weighted loss from outputs O^{t-1} of the previous model $f_{\theta^{t-1}}$.

Cross-entropy loss from the ground truth and pseudo labels. In CSS, the pixels belonging to previous classes become background in the current step. To address this background shift problem, we generate pseudo labels for background pixels using predictions of previous model $f_{\theta^{t-1}}$. The model is trained on the combined ground truth $\tilde{Y}^t$, which consists in labels of current classes and the pseudo labels of all previous classes. Here, $\tilde{Y}^t$ is formulated as

$$\tilde{Y}^t_{i,c} = \begin{cases} 1, & \text{if } Y^t_{i,c_b} = 0 \text{ and } c = \underset{c' \in \mathcal{C}^t}{\text{argmax}} Y^t_{i,c'}, \\ 1, & \text{if } Y^t_{i,c_b} = 1 \text{ and } c = \underset{c' \in \mathcal{C}^{1:t-1}}{\text{argmax}} O^{t-1}_{i,c'}, \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

For non-background pixel i, we copy the ground-truth label from $Y^t_{i,c}$. For background pixel i, we use outputs $O^{t-1}_{i,c}$ of a previous model. When the predictions of old steps are likely to be incorrect, using these predictions as pseudo labels can degrade the performance of the current model. We adopt the practice of [9] to assign $\tilde{Y}^t_{i,c} = 0$ when the uncertainty of pixel i belonging to class c is greater than a certain threshold. Further details are given in [9].

The cross-entropy loss from pseudo labels is formulated by

$$\mathcal{L}_{\text{CE}} = -\frac{\lambda}{HW} \sum_{i=1}^{HW} \sum_{c \in \mathcal{C}^{1:t}} \tilde{Y}^t_{i,c} \log O^t_{i,c}, \quad (4)$$

where λ denotes the percentage of the accepted old class pixels above the certainty threshold over all old class pixels.

Distillation loss from features. When the model features diverge from what it previously learned, catastrophic forgetting occurs. Recent work [9, 25] distill the features of a previous model to reduce this feature divergence problem. Here, we generalize previous methods and introduce a feature knowledge distillation. The feature distillation loss is formulated as

$$\mathcal{L}_{\text{fkd}} = \sum_{l=1}^{L} \left| \Theta(f^t_l(X^t)) - \Theta(f^{t-1}_l(X^t)) \right|^2_2, \quad (5)$$

where $f^t_l(.)$ is the l-th layer in the network M_t and $\Theta(.)$ is a function summarizing spatial statistics of the feature map.

Different choices of the summarizing function $\Theta(.)$ lead to different FKD strategies. For example, pooled outputs distillation (POD) [10] summarizes features across height, width and channel dimensions. Local POD [9], an extension of POD, summarizes features in local regions of different scales. We use local POD as the summarizing function $\Theta(.)$ to capture multi-scale information, which is effective for semantic segmentation.

Class similarity weighted knowledge distillation loss. Our CSW-KD method revises the old knowledge that is more likely to be forgotten, i.e., classes that are visually similar to a new class. We propose to reweigh the predictions O^{t-1} of a previous model on new pixels by the class similarity score S. The reweighted outputs are distilled to the current model when it learns a new class. This allows the model to simultaneously re-learn old classes that are more likely to be forgotten and capture semantic relations between new and old classes.

For every pixel i and its actual new class u, we reweigh $O^{t-1}_{i,v}$, the output of a previous model for the old class v on pixel i, by the similarity between new class u and old class v. Let $P^t \in \mathbb{R}^{U \times C}$ and $\mathcal{P}^{t-1} \in \mathbb{R}^{V \times C}$ respectively denote two sets of C-dimensional prototype vectors of U new classes and all V old classes, where $U = |\mathcal{C}^t|$ and $V = |\mathcal{C}^{1:t-1}|$.

We construct a prototype map $M^t = [\mathbf{m}^t_i] \in \mathbb{R}^{HW \times C}$ where each pixel i contains a prototype vector $\mathbf{m}^t_i = \mathbf{p}^t_c$ based on the pixel label y_i in the segmentation map. Here, class c is the label y_i of pixel i, where $i = 1, 2, ..., HW$. Then we compute a similarity map $S \in \mathbb{R}^{HW \times V}$ between the prototype $\mathbf{m}^t_i$ of a new class in each pixel i and the prototype $\mathbf{p}^{t-1}_v \in \mathcal{P}^{t-1}$ of old class v. Each entry $s_{i,v}$ is the cosine similarity between $\mathbf{m}_i$ and $\mathbf{p}^{t-1}_v$:

$$s_{i,v} = \frac{\mathbf{m}_i \cdot \mathbf{p}^{t-1}_v}{\|\mathbf{m}_i\| \cdot \|\mathbf{p}^{t-1}_v\|}. \quad (6)$$

The similarity map is normalized to reflect the probability that the new class y_i at pixel i is similar to old class v. The normalized similarity map $\tilde{S}$ is defined as

$$\tilde{s}_{i,v} = \frac{\exp s_{i,v}}{\sum_{j=1}^{V} \exp s_{i,j}}. \quad (7)$$

Our CSW-KD method first selects the old classes v that are more likely to be forgotten. It then distills their outputs $O^{t-1}_{i,v}$ weighted by the similarity score $\tilde{S}_{i,v}$ with the new class y_i. We filter out the old classes v that have similarity scores $\tilde{S}_{i,v}$ less than a certain threshold δ. The weighted outputs $\tilde{O}^{t-1}_{i,v}$ are defined as

$$\tilde{O}^{t-1}_{i,v} = \begin{cases} 0, & \text{if } y_i \in \mathcal{C}^{1:t-1} \text{ and } \tilde{S}_{i,v} < \delta, \\ \tilde{S}_{i,v} O^{t-1}_{i,v}, & \text{if } y_i \in \mathcal{C}^t. \end{cases} \quad (8)$$

Here, we set threshold δ based on the total number of old classes $|\mathcal{C}^{1:t-1}|$:

$$\delta = \frac{1}{|\mathcal{C}^{1:t-1}|}. \quad (9)$$

The CSW-KD method distills the weighted outputs $\tilde{O}^{t-1}$ to the current model:

$$\mathcal{L}_{\text{csw-kd}} = -\frac{1}{HW}\sum_{i=1}^{HW}\sum_{c\in\mathcal{C}^{1:t-1}}\tilde{O}_{i,c}^{t-1}\log O_{i,c}^{t}. \quad (10)$$

Via CSW-KD, the current model f_{θ^t} learns the reweighted outputs $\tilde{O}^{t-1}$, and consequently captures the class similarity scores $\tilde{S}$ embedded in $\tilde{O}^{t-1}$. Learning this semantic similarity provides two benefits. First, the model can relate the new class with what it previously learned, thus, transferring the old knowledge for better learning new classes. Second, it encourages the model to implicitly learn the underlying class hierarchy.

Finally, the combined loss is defined as

$$\mathcal{L} = \mathcal{L}_{\text{ce}} + \alpha_1\mathcal{L}_{\text{fkd}} + \alpha_2\mathcal{L}_{\text{csw-kd}}, \quad (11)$$

where α_1 and α_2 denote the weights of each term, which are fine-tuned to find the optimal performance.

Prototype computing. We obtain the prototype of new class c by computing an in-batch average on the logits $Z \in \mathbb{R}^{H\times W\times C}$. Given a batch of logit maps $\mathcal{B} \in \mathbb{R}^{B\times H\times W\times C}$, we flatten out the batch, height and width dimensions and index the logits as z_i, where $i = 1,\ldots,BHW$. The centroid of class c is computed as

$$\mathbf{p}_c = \frac{\sum_{i=1}^{BHW} z_i\,\mathbb{1}[y_i = c]}{|\{i:\ y_i = c\}|}, \quad (12)$$

where $\mathbb{1}[y_i = c] = 1$ if the label y_i is c, and 0 otherwise. The cummulative prototypes $\mathcal{P}^t$ of all classes from task 1 to task t are computed at the end of task t.

4. Experiments

4.1. Experimental setup

Datasets. We perform experiments with REMINDER on two standard image semantic segmentation datasets: Pascal-VOC 2012 [11] and ADE20k [41]. Pascal-VOC 2012 contains 20 foreground classes. Its training and testing sets contain 10,582 and 1,449 images, respectively. ADE20k has 150 foreground classes, 20,210 training images, and 2,000 testing images.

CSS settings. CSS has two experimental settings [2]: disjoint and overlapped. In the disjoint setup, all pixels in the images at each step belong to either the previous classes or the current class. In the overlapped setting, the dataset at each step contains all the images that have pixels of at least one current class, and all pixels from previous and future tasks are labeled as background. We perform experiments in the overlapped setting as this is the most realistic and challenging setting.

For the *Pascal-VOC 2012* dataset, we perform three different experiments: adding 1 class after training with 19 classes (19-1 setting with 2 steps), adding 5 classes all at once after training with 15 classes (15-5 setting with 2 steps), adding 5 classes sequentially after training with 15 classes (15-1 setting with 6 steps).

For the *ADE20k* dataset, we perform four different experiments: adding 50 class after training with 100 classes (100-50 setting with 2 steps), adding 50 classes each time after training with 50 classes (50-50 setting with 3 steps), adding 10 classes each time sequentially after training with 100 classes (100-10 setting with 6 steps), and adding 5 classes each time sequentially after training with 100 classes (100-5 setting with 11 steps).

Metrics. We evaluate the model performance by four mean intersection over union (mIoU) metrics. First, we compute mIoU for the initial classes $\mathcal{C}^0$, which reflects model rigidity: the model resilience to catastrophic forgetting. Second, we compute mIoU for all incremented classes $\mathcal{C}^{1:T}$, which measures plasticity: the model capacity in learning new tasks. Third, we compute mIoU of all classes in $\mathcal{C}^{0:T}$ (*all*), which shows the overall performance of models. Lastly, we report the average of mIoU (*avg*) measured step after step as proposed by [9], which evaluates performance over the entire continual learning process.

Baselines. We benchmark our model against the latest state-of-the-art CSS methods PLOP [9], SDR [24], MiB [2] and ILT [23]. We also evaluate our model against the general continual learning methods: EWC [18] and LwF-MC [31]. For a fair comparison, state-of-the-art methods have been re-trained with a Deeplab-v3 architecture [5] and a ResNet-101 backbone [13].

4.2. Comparisons with the state-of-the-arts

Quantitative evaluation. We compare experimental results of REMINDER with current state-of-the-art methods. For the Pascal-VOC 2012 dataset, Table 1 shows results on the 19-1 (2 tasks), 15-5 (2 tasks) and 15-1 (6 tasks) settings. REMINDER outperforms all other methods on *all* and *avg* mIoU. On the short 15-5 setting (with 2 tasks), our model performs better than PLOP by 1.11% on the *all* mIoU. On the long 15-1 setting (6 tasks), REMINDER improves PLOP by 1.75% on mIoU of new classes (16-20). This shows that our model can learn new knowledge by relating the new with the previously learned concepts. Furthermore, our model outperforms both the recent methods, PLOP and SDR, by 7.07% and 56.46% on the *all* mIoU. REMINDER is more resilient to forgetting than other methods when the model continually learns more tasks.

For the ADE20k dataset, Table 2 shows results on the 100-50 (2 tasks), 50-50 (3 tasks) and 100-10 (6 tasks) settings. On the short 100-50 setting (2 tasks), REMINDER outperforms PLOP by 4.27% and 1.85% on the *all* and *avg* metrics, respectively. On the medium 100-10 setting (6 tasks), REMINDER improves PLOP by a large margin

Table 1. CSS results on Pascal-VOC 2012 in mIoU (%). †: excerpted from [9]. Other results come from our re-implementation.

Method	**19-1** (2 tasks)				**15-5** (2 tasks)				**15-1** (6 tasks)			
	0-19	20	*all*	*avg*	0-15	16-20	*all*	*avg*	0-15	16-20	*all*	*avg*
EWC† [18]	26.90	14.00	26.30		24.30	35.50	27.10		0.30	4.30	1.30	
LwF-MC† [31]	64.40	13.30	61.90		58.10	35.00	52.30		6.40	8.40	6.90	
ILT† [23]	67.75	10.88	65.05	71.23	67.08	39.23	60.45	70.37	8.75	7.99	8.56	40.16
MiB [2]	70.57	22.82	68.30	72.95	75.30	48.68	68.96	75.07	39.47	14.50	33.53	54.44
SDR [24]	68.52	23.29	66.37	71.48	75.21	46.72	68.64	74.32	43.08	19.31	37.42	54.52
PLOP [9]	75.50	30.22	73.35	75.43	75.44	49.65	69.30	74.82	63.41	26.76	54.68	66.96
REMINDER	**76.48**	**32.34**	**74.38**	**76.22**	**76.11**	**50.74**	**70.07**	**75.36**	**68.30**	**27.23**	**58.52**	**68.27**
Joint	77.45	77.94	77.47		78.88	72.63	77.39		78.88	72.63	77.39	

Table 2. CSS results on ADE20k in mIoU (%). †: excerpted from [9].

Method	**100-50** (2 tasks)				**50-50** (3 tasks)				**100-10** (6 tasks)			
	0-100	101-150	*all*	*avg*	0-50	51-150	*all*	*avg*	0-100	101-150	*all*	*avg*
ILT† [23]	18.29	14.40	17.00	29.42	3.53	12.85	9.70	30.12	0.11	3.06	1.09	12.56
MiB [2]	40.52	17.17	32.79	37.31	45.57	**21.01**	29.31	38.98	38.21	11.12	29.24	35.12
SDR [24]	40.52	17.17	32.79	37.31	45.66	18.76	27.85	34.25	37.26	12.13	28.94	34.48
PLOP [9]	**41.76**	14.52	32.74	37.73	**47.33**	20.27	**29.41**	38.75	38.59	14.21	30.52	34.48
REMINDER	41.55	**19.16**	**34.14**	**38.43**	47.11	20.35	29.39	**39.26**	**38.96**	**21.28**	**33.11**	**37.47**
Joint	44.34	28.21	39.00		51.21	32.77	39.00		44.34	28.21	39.00	

Table 3. CSS results in mIoU (%) on ADE20k 100-5 setting.

Method	**100-5** (11 tasks)			
	0-100	101-150	*all*	*avg*
ILT† [23]	0.08	1.31	0.49	7.83
MiB† [2]	36.01	5.66	25.96	32.69
SDR [24]	33.02	10.63	25.61	33.07
PLOP [9]	35.72	12.18	27.93	35.10
REMINDER	**36.06**	**16.38**	**29.54**	**36.49**

of 8.49% on *all* and 8.67% on *avg*. On 50-50 setting, REMINDER is on par with PLOP with a slight decrease on the *all* metric, while outperforming PLOP by 1.31% on the *avg* metric measured across all tasks.

Table 3 compares the model performance on the longest 100-5 setting with 11 tasks. Our REMINDER performs better than PLOP by 5.76% and 3.96% on *all* and *avg*, respectively. Between prototype-based approaches, REMINDER outperforms SDR by a large margin 15.34% in *all* mIoU. Notably, the proposed REMINDER significantly outperforms PLOP by 34.48% on newly learned classes (i.e., class 101-150). REMINDER yields the better rigidity-plasticity trade-off than the competitors, especially in long CL settings.

Table 4 shows the per-class mIoU of different methods on the Pascal-VOC 15-1 setting. The model learns *tv* at the last step. Our REMINDER consistently outperforms previous methods on the *object* class. Via REMINDER, the model remembers better visually similar classes. Furthermore, REMINDER also outperforms other contenders on a newly learned *tv* class. This shows that relating old and new concepts assists the learning new knowledge.

Qualitative evaluation. Visualization results of segmentation maps of REMINDER, PLOP, and MiB are shown in Fig. 3 on two test images of Pascal-VOC 2012. For the first image (Row 1-3), PLOP and MiB gradually forget class *dog* from Step 3. The model gets confused between visually similar images. Compared with other methods, REMINDER better distinguishes between two similar animal classes, *dog* and *sheep*. Our framework selectively reminds the model of old class *dog* as it learns *sheep*. Thus, the model's knowledge on *dog* is less affected when learning the visually similar class *sheep*. For the second image (Row 4-6), when learning too similar new class (i.e., *sheep*) at Step 3, our REMINDER still retains small parts of *cow*, while other methods completely forget *cow*. In Step 4-6, PLOP gets confused *cow* with *horse*, while REMINDER retains most part of its correct prediction on *cow*. Since CSW-KD enforces the model to learn how similar two classes are, we hypothesize that the model can detect more subtle differences between similar concepts.

4.3. Ablation study

Effectiveness of class hierarchy learning. We explore the model's ability to learn class hierarchy by visualizing feature distribution by class. Fig. 4 visualizes t-SNE distribution of features extracted from our REMINDER and

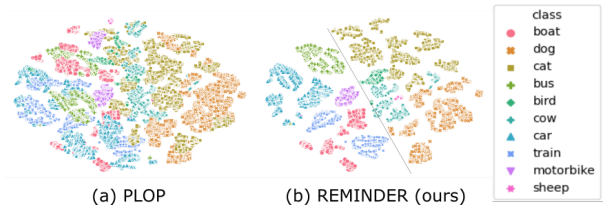
Figure 3. Visualization results of MiB, PLOP and REMINDER across 6 steps of the 15-1 setting of CSS for two test images in Pascal-VOC 2012. On rows 1-3, MiB and PLOP are confused betweed *dog* and *sheep* at Step 3, *person* and *sofa, train* at Steps 4-5, while REMINDER suffers much less. On rows 4-6, REMINDER is less confused between *cow* and *horse*, *sheep* compared to PLOP and MiB.

Table 4. Per-class IoU on the Pascal-VOC 15-5 setting.

$M_0(20)$	Base classes																New classes					*all*
	backgr.	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motorbike	person	pot. plant	sheep	sofa	train	tv	
MiB [2]	84.59	11.30	16.18	37.50	26.03	49.48	6.26	41.08	75.03	1.24	33.34	36.95	64.56	44.04	23.53	80.46	0.84	24.65	14.69	15.21	17.12	33.52
SDR [24]	82.00	16.65	19.54	18.82	0.88	21.46	32.70	35.06	47.81	12.66	4.06	8.05	37.42	32.96	12.74	74.55	15.44	9.66	9.66	11.47	9.77	24.44
PLOP [9]	80.16	66.09	27.11	47.02	52.47	62.82	83.94	80.70	80.43	33.56	64.82	55.24	75.46	62.97	75.04	66.79	**20.95**	**49.44**	18.02	31.58	13.82	54.68
REMINDER (Ours)	**85.77**	**73.58**	**32.50**	**65.10**	**59.58**	**67.45**	**85.64**	**82.99**	**84.91**	**34.55**	**67.64**	**57.74**	**79.22**	**70.90**	**76.70**	**68.58**	16.26	46.86	**18.79**	**36.58**	**17.65**	**58.52**

PLOP [9]. Features of REMINDER are well-clustered and yield a separation between animals and vehicles. Our CSW-KD method enables the model to capture the semantic similarity between classes and implicitly learn the underlying hierarchy. The well-separated feature distribution also shows that our model can extract discriminative representations and distinguish visually similar classes better.

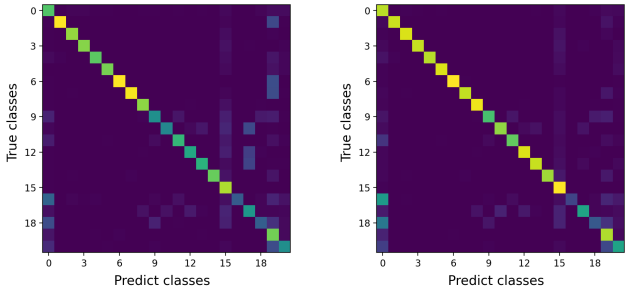
Reducing the forgetting of similar old classes. We



(a) PLOP (b) REMINDER (ours)

Figure 4. Class hierarchy learned by REMINDER. T-SNE visualization of features learned by PLOP and REMINDER.

investigate our model's efficiency in reducing the forgetting of old classes similar to the new one. Fig. 5 shows the confusion matrix from the predictions of PLOP [9] and REMINDER on Pascal-VOC 15-1 setting. PLOP misjudges the old vehicle classes - bus (class 6) and car (class 7) - as new class train (class 19). It also mispredicts old animal classes - cow (class 10) and dog (class 12) - as new class sheep (class 17). Our REMINDER distinguishes these similar classes better.



(a) PLOP (b) REMINDER (ours).

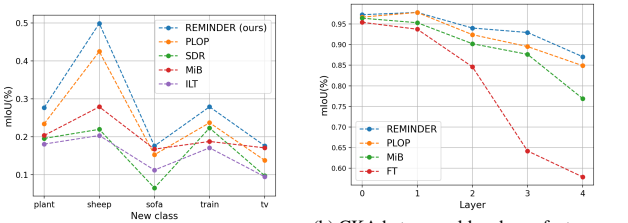
Figure 5. Confusion matrix of (a) PLOP and (b) REMINDER on the Pascal-VOC 15-1 setting.

Impact of each loss objective. We investigate the impact of different distillation loss objectives on the Pascal-VOC 15-1 setting, as shown in Table 5. We apply a feature knowledge distillation (using local POD) with one of the three output knowledge distillation objectives: (i) our proposed CSW-KD, (ii) the normal knowledge distillation (KD), and (iii) the unbiased knowledge distillation (UNKD) in MiB [2]. Our CSW-KD consistently outperforms the UNKD in all settings.

Table 5. Performance of REMINDER on the Pascal-VOC 15-1 setting when using different output distillation losses.

Distillation loss	0-15	16-20	*all*	*avg*
Knowledge Distillation	29.72	4.42	23.69	49.18
UNKD [2]	59.67	20.26	50.29	62.47
CSW-KD	**68.30**	**27.23**	**58.52**	**68.27**

Effectiveness of learning new classes. We examine the model efficiency on learning new tasks on the Pascal-VOC 15-1 setting. When the model learns a new class, we record its performance on that class, as shown in Fig. 6a. The proposed REMINDER (blue curve) achieves the highest mIoU on all new classes. We conjecture that learning the class similarity enforces the model to identify common features between similar classes, thus better transferring the old knowledge to learn new tasks more effectively.



(a) mIoU on a new class (b) CKA between old and new features on different layers.

Figure 6. (a): Model performance on newly learned class. (b): Similarity between features before and after learning all 5 new tasks on the Pascal-VOC 15-1 setting (6 tasks in total).

Feature reuse of REMINDER. We investigate the model's ability to reuse features in REMINDER. Following recent studies [26, 30], the centered kernel alignment (CKA) metric is used to measure the similarity of model representations. We compute similarities between features before and after the model learns all 5 new tasks on the Pascal-VOC 15-1 setting. As shown in Fig. 6b, a normal fine-tuning method (red curve) erases features in the deeper layers of the model, which indicates the forgetting problem. Our REMINDER (blue curve) encourages the most feature reuse. This shows that the model with CSW-KD retains the most knowledge in a long continual learning setting.

5. Conclusions

This paper proposes a new class similarity weighted knowledge distillation (CSW-KD) method to alleviate the forgetting of visually similar classes in continual semantic segmentation. REMINDER - our proposed framework - uses CSW-KD to selectively revise old classes that are more likely to be forgotten. Evaluated on Pascal-VOC 2012 and ADE20k datasets, REMINDER outperforms recent state-of-the-arts methods in both reducing the forgetting of old tasks and promoting the learning of new tasks.

References

[1] Jose Manuel Alvarez, Theo Gevers, Yann LeCun, and Antonio M. Lopez. Road scene segmentation from a single image. In *European Conference on Computer Vision*, pages 376–389, 2012. 1

[2] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulò, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9230–9239, 2020. 1, 2, 5, 6, 7, 8

[3] Fabio Cermelli, Massimiliano Mancini, Yongqin Xian, Zeynep Akata, and Barbara Caputo. Prototype-based incremental few-shot semantic segmentation. In *British Machine Vision Conference*, pages 484–498, 2021. 3

[4] Arslan Chaudhry, Puneet Kumar Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision*, pages 556–572, 2018. 2

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arxiv*, abs/1706.05587, 2017. 5

[6] Zhijie Deng, Yucen Luo, and Jun Zhu. Cluster alignment with a teacher for unsupervised domain adaptation. In *IEEE International Conference on Computer Vision*, pages 9943–9952, 2019. 2, 3

[7] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5133–5141, 2019. 2

[8] Nanqing Dong and Eric P. Xing. Few-shot semantic segmentation with prototype learning. In *British Machine Vision Conference*, pages 79–93, 2018. 3

[9] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. PLOP: Learning without forgetting for continual semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4050, 2021. 1, 2, 4, 5, 6, 7, 8

[10] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *European Conference on Computer Vision*, pages 86–102, 2020. 4

[11] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88:303–338, 2009. 5

[12] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999. 1

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 5

[14] Yufan He, Dong Yang, Holger Roth, Can Zhao, and Daguang Xu. DiNTS: Differentiable neural network topology search for 3d medical image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5841–5850, 2021. 1

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, pages 1–9, 2015. 3

[16] Zilong Huang, Wentian Hao, Xinggang Wang, Ming Tao, Jianqiang Huang, Wenyu Liu, and Xiansheng Hua. Half-real half-fake distillation for class-incremental semantic segmentation. *ArXiv*, abs/2104.00875, 2021. 2

[17] Joel Janai, Fatma Guney, Aseem Behl, and Andreas Geiger. Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art. *Foundations and Trends in Computer Graphics and Vision*, 12:1–308, 2020. 1

[18] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114:3521 – 3526, 2017. 5, 6

[19] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, pages 3925–3934, 2019. 2

[20] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018. 2

[21] Andrea Maracani, Umberto Michieli, Marco Toldo, and Pietro Zanuttigh. Recall: Replay-based continual learning in semantic segmentation. In *IEEE International Conference on Computer Vision*, pages 7026–7035, 2021. 2, 3

[22] James L McClelland, Bruce L McNaughton, and Andrew K Lampinen. Integration of new information in memory: new insights from a complementary learning systems perspective. *Philosophical Transactions of the Royal Society B*, 375(1799):20190637, 2020. 2

[23] Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. *International Conference on Computer Vision Workshop*, pages 3205–3212, 2019. 1, 2, 5, 6

[24] Umberto Michieli and Pietro Zanuttigh. Continual semantic segmentation via repulsion-attraction of sparse and disentangled latent representations. In *IEEE Conference on Conference on Computer Vision and Pattern Recognition*, pages 1114–1124, 2021. 1, 2, 3, 5, 6, 7

[25] Umberto Michieli and Pietro Zanuttigh. Knowledge distillation for incremental learning in semantic segmentation. *Computer Vision and Image Understanding*, 205:103167, 2021. 4

[26] Ari S Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing System*, pages 5732–5741, 2018. 8

[27] Firat Ozdemir, Philipp Fuernstahl, and Orcun Goksel. Learn the new, keep the old: Extending pretrained models with new anatomy and images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 361–369, 2018. 1

[28] Firat Ozdemir and Orcun Goksel. Extending pretrained segmentation networks with additional anatomical structures. *International Journal of Computer Assisted Radiology and Surgery*, 14(7):1187–1195, 2019. 1

[29] Pedro H. O. Pinheiro. Unsupervised domain adaptation with similarity learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8004–8013, 2018. 3

[30] Vinay V Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. In *International Conference on Learning Representation*, 2021. 8

[31] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, G. Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5533–5542, 2017. 5, 6

[32] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557, 2018. 2

[33] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Conference on Neural Information Processing Systems*, page 2994–3003, 2017. 2

[34] Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998. 1

[35] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *IEEE International Conference on Computer Vision*, pages 9196–9205, 2019. 3

[36] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019. 1

[37] Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, Yuxin Peng, and Zheng Zhang. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *ACM International Conference on Multimedia*, pages 177—-186, 2014. 2

[38] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. Learning semantic representations for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 5423–5432, 2018. 3

[39] Shipeng Yan, Jiale Zhou, Jiangwei Xie, Songyang Zhang, and Xuming He. An EM framework for online incremental learning of semantic segmentation. In *ACM International Conference on Multimedia*, pages 3052–3060, 2021. 2, 3

[40] Qihang Yu, Dong Yang, Holger Roth, Yutong Bai, Yixiao Zhang, Alan L Yuille, and Daguang Xu. C2fnas: Coarse-to-fine neural architecture search for 3d medical image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4126–4135, 2020. 1

[41] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5122–5130, 2017. 5