SSUL: Semantic Segmentation with Unknown Label for Exemplar-based Class-Incremental Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

We consider a class-incremental semantic segmentation (CISS) problem. While some recently proposed algorithms utilized variants of knowledge distillation (KD) technique to tackle the problem, they only partially addressed the key additional challenges in CISS that causes the catastrophic forgetting; *i.e.*, the semantic drift of the background class and multi-label prediction issue. To better address these challenges, we propose a new method, dubbed as SSUL-M (Semantic Segmentation with Unknown Label with Memory), by carefully combining several techniques tailored for semantic segmentation. More specifically, we make three main contributions; (1) modeling *unknown* class within the background class to help learning future classes (help plasticity), (2) *freezing* backbone network and past classifiers with binary cross-entropy loss and pseudo-labeling to overcome catastrophic forgetting (help stability), and (3) utilizing *tiny exemplar memory* for the first time in CISS to improve *both* plasticity and stability. As a result, we show our method achieves significantly better performance than the recent state-of-the-art baselines on the standard benchmark datasets. Furthermore, we justify our contributions with thorough and extensive ablation analyses and discuss different natures of the CISS problem compared to the standard class-incremental learning for classification.

1 Introduction

Class incremental learning (CIL) problem, in which a learner should incrementally learn newly arriving class objects while not "catastrophically" forgetting the past learned classes, is one of the fundamental, yet still open, problem in machine learning. Since the seminal work, [25], most of the recent neural network-based CIL research has focused on the classification setting, and various approaches have been proposed to address the main challenge of the problem, the so-called plasticity-stability dilemma, *e.g.*, [19, 28, 12, 1, 2, 8], to name a few.

More recently, the CIL framework has been extended to more complex semantic segmentation tasks [23, 3, 7], motivated by the practical need in various applications such as autonomous driving, etc. One of the key additional difficulties of the class-incremental semantic segmentation (CISS) problem lies in the semantic drift of the background class that is present in the incrementally arriving training data. Namely, the label "background (BG)" is assigned to all the pixels that are not the *current* class object in the data, and the pixels could belong to *three* different categories: future object classes that are not yet seen by the model, past object classes that are already learned, and the true background.

A few recent work attempted to address above semantic drift issue by leveraging and modifying the knowledge-distillation (KD) [19] technique, which has been popular for CIL in standard image classification. Namely, the initiative study [23] modified the KD for CISS in a straightforward way, and [3] proposed a strategy to incorporate the BG class probability in computing the cross-entropy and distillation losses. Furthermore, [7], the current state-of-the-art, utilized the pseudo-labeling of the

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

BG pixels of the current task data with previous task's model for the cross-entropy loss, together with the multi-scale feature distillation scheme adopted from [8]. However, we argue that above previous work only partially addressed the BG label issue; namely, [3] naively *added* the class probabilities to modify the cross-entropy and distillation losses, which makes it hard to have fine-grained learning of prediction probability for each class, each pixel. Moreover, [7] could only handle the case of BG class being assigned to the pixels for the *past* classes via pseudo-labeling and lacked any mechanism for handling the case of *future* classes being labeled as the BG class. Consequently, the CIL performance of their methods, measured by mean Intersection-over-Union (mIoU), has been largely unsatisfactory, *i.e.*, showing large gap with the joint training upper bound.

In this paper, we first identify that the multi-label prediction of semantic segmentation poses another key challenge of CISS and propose SSUL-M (Semantic Segmentation with Unknown Label with Memory) that much better addresses these challenges. More specifically, we make the following three main contributions. Firstly, we introduce additional "*Unknown*" class label that is assigned to the objects that are present in the background, detected by the off-the-shelf saliency-map detector. By augmenting the BG label with this additional class, we let the base feature extractor distinguish the representations of the potential future class objects and the true background. Secondly, we adopt the pseudo-labeling strategy as in [7] and further augment the BG & Unknown class labels with the past class labels, but with two important differences in concrete learning strategies to stabilize the classification scores and improve the *precision* of the prediction. One is that we use the separate *sigmoid*, instead of the softmax, output classifier for each class so that the logit score can be learned in an absolute sense per class. The other is that we *freeze* the base feature extractor and the classifiers for past classes after initial learning, to strictly maintain the knowledge of the past classes. Thirdly, we utilize an *exemplar memory* that stores a tiny portion of training data for past classes to use them as anchors and further improve the mIoU. Note that using the exemplar memory is a standard practice for CIL in classification, but it has been overlooked in CISS so far. Moreover, we show that the memory mainly helps to improve the mIoU for the *current* classes, in contrast to a common belief that it mainly is a tool for not forgetting past classes.

With above contributions, our SSUL-M achieved the strong state-of-the-art performance on popular benchmarks with *significantly large* margin over the recent baselines in [3, 7], particularly when the number of incremental tasks gets larger. Furthermore, we carry out extensive ablation study as well as both quantitative and qualitative analyses to convincingly highlight the strength of our method.

2 Related Work

Class incremental learning (CIL) As a branch of general continual learning [26, 24], CIL considers the setting in which tasks with new class objects arrive incrementally and the model needs to classify *all* the classes observed so far jointly. It is well known that neural network-based CIL suffers from catastrophic forgetting [22], mainly caused by the score bias toward the new classes due to the training data imbalance. Most of the CIL research has been focused on the classification tasks, and the exemplar-memory based methods combined with KD [28, 12, 1, 2, 8] are shown to achieve the state-of-the-art performance by re-balancing the biased predictions of the classifier.

Class incremental semantic segmentation (CISS) Contrary to the current trend of CIL for classification, studies for semantic segmentation so far have only focused on the setting which do not utilize the exemplar-memory [23, 3, 7]. [3] first explored to address the issue of semantic drift of BG label, and [7] proposed to use pseudo-labels to augment the BG label, all without using exemplar-memory. To the best of our knowledge, we propose to use exemplar-memory as well as handle potential future classes in BG label for the first time for CISS.

Saliency map detection Salient object detection is a fundamental computer vision task that identifies the most visually distinctive objects in an image, and we use the off-the-shelf saliency-map detector to define the *Unknown* class in the BG label. The early salient object detection method, DRFI [13], conducts a multi-level segmentation with a random forest regressor to detect a salient object. Recent deep neural network based method exploits rich multi-scale feature maps with short connections [11] and pooling-based modules [20]. The saliency information has widely been utilized on various tasks. For example, the weakly-supervised semantic segmentation approaches [27, 29, 14] generate pseudo-labels filtering out the background regions using saliency map, and recent data augmentation approaches [16, 15] also exploit the saliency information to find the optimal mixing of mask.

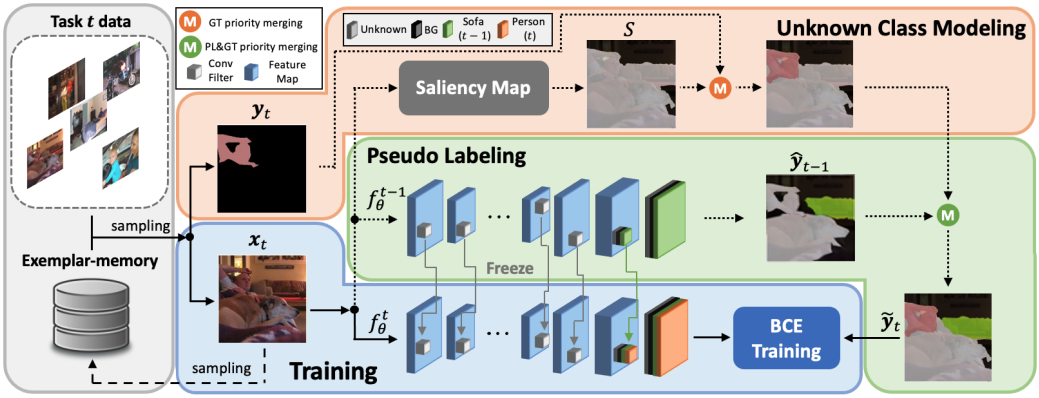


Figure 1: Overall procedure of SSUL. Given $(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathcal{D}_t$, the augmented label $\tilde{\boldsymbol{y}}_t$ is first obtained by the "Unknown" class modeling and pseudo-labeling. Then, using $\tilde{\boldsymbol{y}}_t$ as a target, we update $f_{\boldsymbol{\theta}}^t$ with model freezing and BCE loss. The exemplar-memory is also updated with class-balanced sampling.

3 Main Method

3.1 Notations and Problem Setting

We consider exactly the same setting as considered in [3, 7]. In CISS, the learning happens with $t = 1, \dots, T$ incremental tasks. For each incremental state t, we observe a training dataset $\mathcal{D}_t$ that consists of pairs $(\boldsymbol{x}_t, \boldsymbol{y}_t)$, in which $\boldsymbol{x}_t \in \mathcal{X}$ denotes an input image of size N, and $\boldsymbol{y}_t \in \mathcal{Y}_t^N$ denotes the corresponding ground-truth (GT) *pixel* labels. The label space $\mathcal{Y}_t = \{c_b\} \cup \mathcal{C}_t$ consists of the current classes in task t, $\mathcal{C}_t$, and the dummy background class c_b, that is assigned to all pixels that do not belong to $\mathcal{C}_t$. Thus, the c_b label can be assigned to the objects with the *past* classes $\mathcal{C}^{1:t-1}$, the objects with the *future* classes $\mathcal{C}^{t+1:T}$, or the true background pixels.

After learning task t, the semantic segmentation model $f_{\boldsymbol{\theta}}^t$ is required to predict whether a pixel belongs to *all* the classes learned so far, $\mathcal{C}^{1:t} = \mathcal{C}^{1:t-1} \cup \mathcal{C}_t$, or the true background. As in other work, we assume the classes in each $\mathcal{C}_t$ are disjoint. Typically, the model is defined to be a mapping $f_{\boldsymbol{\theta}}^t : \mathcal{X} \to \mathbb{R}^{N \times |\mathcal{Y}^t|}$, in which $\mathcal{Y}^t = \mathcal{C}_d \cup \mathcal{C}^{1:t}$ with $\mathcal{C}_d$ containing dummy labels. All previous work [3, 7] simply set $\mathcal{C}_d = \{c_b\}$, but in our work, we also add the separate "*Unknown*" class label, c_u, to $\mathcal{C}_d$, hence, set $\mathcal{C}_d = \{c_b, c_u\}$. Later, we show defining this additional label c_u in our model output plays a critical role in improving the learning capability for future classes. The architecture of $f_{\boldsymbol{\theta}}^t$ is typically a fully-convolutional network, which consists of a convolutional feature extractor, $h_{\boldsymbol{\psi}}^t$, followed by the final 1×1 classifier filters, $\{\boldsymbol{\phi}_c^t\}_{c \in \mathcal{Y}^t}$, one for each output class in $\mathcal{Y}^t$.

The learning of $f_{\boldsymbol{\theta}}^t$ is done in conjunction with the previous model $f_{\boldsymbol{\theta}}^{t-1} : \mathcal{X} \to \mathbb{R}^{N \times |\mathcal{Y}^{t-1}|}$ to prevent forgetting during incrementally updating the model. Determining which output classifier (*e.g.*, softmax or sigmoid) to use for each pixel as well as how to transfer the knowledge of $f_{\boldsymbol{\theta}}^{t-1}$ to $f_{\boldsymbol{\theta}}^t$ (*e.g.*, knowledge distillation or model freezing) are design choices, and we elaborate our choices more in details in the subsequent sections. Furthermore, we denote $\mathcal{M}$ as the *exemplar memory*, which can store a small number of samples from past training data, $\mathcal{D}^{1:t-1}$, and use it for learning $f_{\boldsymbol{\theta}}^t$.

Once the learning of $f_{\boldsymbol{\theta}}^t$ is done, the prediction for pixel i of an input image $\boldsymbol{x}$ is obtained by

$$\hat{\boldsymbol{y}}_i^t = \arg\max_{c \in \mathcal{Y}^t} f_{\boldsymbol{\theta},ic}^t(\boldsymbol{x}),$$

and the performance is measured by the mean intersection-over-union (mIoU) metric for the classes in $\mathcal{Y}^t$. (Only in the evaluation phase, we merge c_b and c_u for computing mIoU of the BG class.)

3.2 Two Additional Challenges of CISS

In addition to the typical reason for causing catastrophic forgetting in CIL for classification, *i.e.*, the prediction bias due to the data imbalance, we note there are two additional unique challenges in CISS.

The first challenge, as mentioned in the Introduction and in previous work [3, 7], comes from the semantic drift of the background (BG) label. Namely, the ground-truth label associated with the

pixels of the object can change depending on the incremental state. For example, when a pixel is labeled as "BG" at state t, it is possible that the corresponding pixel would be labeled as " sofa" at state $t-1$ (*i.e.,* past class) or labeled as "dog" at state $t+1$ (*i.e.*, future class), depending on the true object to which the pixel belongs. Therefore, naively learning with the "BG" target label for the pixel at state t could cause either severe forgetting of the past class (*i.e.*, hurting stability) or interfering the learning of the future class (*i.e.*, hurting plasticity).

The second challenge stems from the fact that semantic segmentation is a multi-label prediction problem. Namely, for a given image, the segmentation model needs to output a *set* of classes, in contrast to the classification model which outputs only a single class. Therefore, the *precision* of the prediction for each pixel becomes important in addition to the recall as is reflected in the mIoU performance metric; *i.e.*, not only predicting a correct class for a pixel is important, but also not predicting a wrong class is important for the overall metric. That is, if the prediction for every pixel gets biased toward the current classes in $\mathcal{C}_t$, the mIoU's for the past classes in $\mathcal{C}^{1:t-1}$ *as well as* the current classes would significantly drop jointly. This point is exactly why even the mIoU's for the newly learned classes are *very low* in [3, 7]. Note this is in a stark difference with the classification, in which the accuracy of the current classes would remain high even with the severe bias and forgetting, since it is a single-label prediction problem.

To address above unique challenges of CISS, we devise our SSUL-M by carefully combining several ideas, of which overall procedure is outlined in Figure 1. We now elaborate on our three main contributions in details: 1) **label augmentation** for BG class with *Unknown* class and pseudo-labels, 2) **stable score learning** with model freeze and sigmoid output, and 3) usage of **tiny exemplar memory** with class-balanced sampling.

3.3 Contribution 1: Label Augmentation for BG Class

Here, we describe how we generate an augmented target label $\tilde{\boldsymbol{y}}_t \in (\mathcal{Y}^t)^N = \{c_b, c_u, \mathcal{C}^{1:t}\}^N$ given a training sample $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ in $\mathcal{D}_t$. Recall that $\boldsymbol{y}_t \in (\mathcal{Y}_t)^N = \{c_b, \mathcal{C}_t\}^N$.

Unknown class modeling In order to handle the case in which the potential future class objects are labeled as BG, we propose to use a novel unknown class label, c_u, that is defined to designate any foreground objects that are not the current classes in $\mathcal{C}_t$. Specifically, as depicted in the top part of Figure 1, we first apply an off-the-shelf salient object detector [11] $S : \mathcal{X} \to \{0,1\}^N$ to the input image $\boldsymbol{x}_t$, which assigns 1 to the pixels if they are salient (*i.e.*, part of a foreground object) and 0 otherwise. Then, we assign c_u to the pixels that are BG-labeled but salient; namely, we set $\tilde{\boldsymbol{y}}_{t,i} = c_u$ if $(\boldsymbol{y}_{t,i} = c_b) \wedge (S(\boldsymbol{x}_t)_i = 1)$ for pixel i. In our experiments, we show this Unknown class label plays a critical role so that the base feature extractor can distinguish the representations of the potential future class objects and the true background, even before observing the class labels.

Pseudo-labeling Once augmenting with c_u is done, we further augment with pseudo-labels generated from the previous model $f^{t-1}_{\boldsymbol{\theta}}$ to maintain the knowledge of the past classes, similarly as in [7] and as shown in the middle part of Figure 1. Namely, we denote $\hat{\boldsymbol{y}}_{t-1,i} = \arg\max_{c \in \mathcal{Y}^{t-1}} f^{t-1}_{\boldsymbol{\theta},ic}(\boldsymbol{x}_t)$ as the prediction of $f^{t-1}_{\boldsymbol{\theta}}$ for the i-th pixel and set $\tilde{\boldsymbol{y}}_{t,i} = \hat{\boldsymbol{y}}_{t-1,i}$ if

$$(\boldsymbol{y}_{t,i} = c_b) \wedge (\hat{\boldsymbol{y}}_{t-1,i} \in \mathcal{C}^{1:t-1}) \wedge (\boldsymbol{\mu}_i > \tau),$$

in which $\boldsymbol{\mu}_i = \max_{c \in \mathcal{C}^{1:t-1}} \sigma(f^{t-1}_{\boldsymbol{\theta},ic}(\boldsymbol{x}_t))$ stands for the confidence of prediction for $\hat{\boldsymbol{y}}_{t-1,i} \in \mathcal{C}^{1:t-1}$ where $\sigma(\cdot)$ is the sigmoid function, and τ is a threshold (set to 0.5). In words, we assign the pseudo-labels (*i.e.*, the predictions from the previous model) to the pixels that are BG-labeled only when the predictions are made to be the past object classes (excluding c_b and c_u) with enough confidence.

In summary, at the incremental state t, the augmented target label for the i-th pixel becomes

$$\tilde{\boldsymbol{y}}_{t,i} = \begin{cases} \boldsymbol{y}_{t,i} & \text{if } \boldsymbol{y}_{t,i} \in \mathcal{C}_t \\ \hat{\boldsymbol{y}}_{t-1,i} & \text{if } (\boldsymbol{y}_{t,i} = c_b) \wedge (\hat{\boldsymbol{y}}_{t-1,i} \in \mathcal{C}^{1:t-1}) \wedge (\boldsymbol{\mu}_i > \tau) \\ c_u & \text{if } (\boldsymbol{y}_{t,i} = c_b) \wedge (S(\boldsymbol{x}_t)_i = 1) \wedge \{(\hat{\boldsymbol{y}}_{t-1,i} \in \mathcal{C}_d) \vee (\boldsymbol{\mu}_i \leq \tau)\} \\ c_b & \text{else,} \end{cases} \tag{1}$$

in which the pseudo-label is generated only for the incremental state $t \geq 2$. Figure 1 shows a concrete example of augmentating $\boldsymbol{y}_t$ to $\tilde{\boldsymbol{y}}_t$, in which $\mathcal{C}_{t-1} = \{\texttt{sofa}\}$ and $\mathcal{C}_t = \{\texttt{person}\}$.

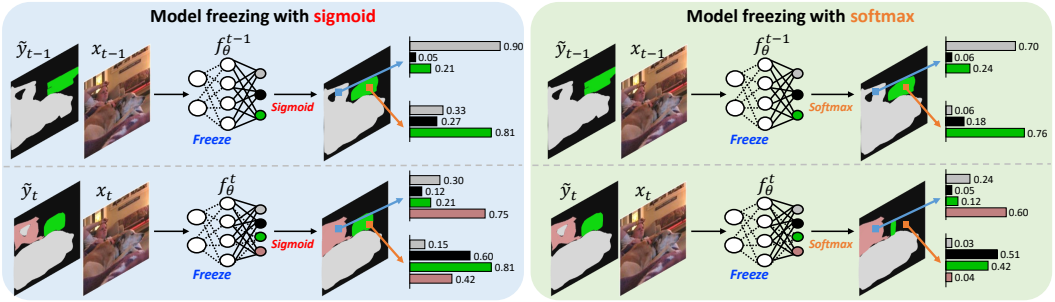


Figure 2: Comparison of the effect on the classification scores for sigmoid with binary cross-entropy (BCE) and softmax with cross-entropy (CE). For a pixel with noisy label $\tilde{\boldsymbol{y}}_{t,i} = c_b$ (*i.e.,* BG label (black) when the true is "sofa" (green)), even when the classifier for "sofa" ($\{\boldsymbol{\phi}_c\}_{c\in\mathcal{C}^{1:t-1}}$) is frozen, the softmax would cause the BG class score to rise above the score for "sofa", whereas the sigmoid would let the BG score rise only moderately. Thus, the classification for the pixel could still become "sofa" for sigmoid, whereas softmax would make a false prediction as BG.

3.4 Contribution 2: Stable Score Learning

We argue that simply using $\tilde{\boldsymbol{y}}_t$ as a target and training $f^t_{\boldsymbol{\theta}}$ with softmax output layer would make the output scores too unstable as the incremental learning continues. The reason is because the augmented labels are *noisy* and the softmax computes the prediction probability in a *relative* way. As shown in our experiments, such instability significantly hurts the precision of the multi-label prediction. To that end, we propose the following three strategies for the stable learning of output scores.

Model freezing Instead of updating the full model with $\tilde{\boldsymbol{y}}_t$ at every state t, we *freeze* the convolutional feature extractor, $h_{\boldsymbol{\psi}}$, after initial learning ($t = 1$) as well as the classifiers for the past classes, $\{\boldsymbol{\phi}_c\}_{c\in\mathcal{C}^{1:t-1}}$, and only learn the 1×1 filters $\boldsymbol{\phi}^t_{c_b}$, $\boldsymbol{\phi}^t_{c_u}$ and $\{\boldsymbol{\phi}^t_c\}_{c\in\mathcal{C}_t}$. Such strict model freezing certainly can prevent forgetting, but is counter-intuitive from the plasticity viewpoint. However, thanks to the unknown class c_u, it turns out the feature extractor $h_{\boldsymbol{\psi}}$ can roughly learn the representations for the potential future class objects present with BG label in $\mathcal{D}_1$. Thus, it becomes sufficient to learn the decision boundaries for c_b, c_u, and $\mathcal{C}_t$ on those representations at state t.

Sigmoid with binary cross entropy loss For learning $\boldsymbol{\phi}^t_{c_b}$, $\boldsymbol{\phi}^t_{c_u}$ and $\{\boldsymbol{\phi}^t_c\}_{c\in\mathcal{C}_t}$, the choice of output layer and loss function becomes important to make the output score stable. A common choice is the softmax with cross-entropy loss, however, since the target labels for c_b and c_u in $\tilde{\boldsymbol{y}}_t$ are noisy, we observe the relative scoring of softmax could cause significant forgetting of past classes $\mathcal{C}^{1:t-1}$. To see this, let $s_{ic} = [f^t_{\boldsymbol{\theta}}(\boldsymbol{x}_t)]_{ic}$ be the score for class c at pixel i. When the softmax with cross-entropy (CE) is used, the gradient of the loss at s_{ic} becomes $\partial\mathcal{L}^t_{\texttt{CE}}(\boldsymbol{\theta})/\partial s_{ic} = \boldsymbol{p}_{ic} - \mathbb{1}\{c = \tilde{\boldsymbol{y}}_{t,i}\}$, in which $\boldsymbol{p}_{ic} = \exp(s_{ic})/(\sum_{c'}\exp(s_{ic'}))$. The issue occurs when $\tilde{\boldsymbol{y}}_{t,i} = c_b$ or c_u, while the true label for the pixel should be the past classes in $\mathcal{C}^{1:t-1}$. (Thus, the pseudo-label $\hat{\boldsymbol{y}}_{t-1,i}$ missed the pixel, which could often happen.) Then, the gradient descent learning will force s_{ic_b} or s_{ic_u} to become much higher than $\{s_{ic}\}_{c\in\mathcal{C}^{1:t-1}}$, the scores obtained from the *frozen* classifiers for past classes. Thus, at test time, for a similarly confusing pixel, the model would tend to predict as c_b or c_u, hence, causing the forgetting of past classes even though their classifiers are fixed.

Therefore, we instead use the sigmoid output with binary cross-entropy (BCE) loss independently for each class. In that case, the gradient of the loss at s_{ic} becomes $\partial\mathcal{L}^t_{\texttt{BCE}}(\boldsymbol{\theta})/\partial s_{ic} = \sigma(s_{ic}) - \mathbb{1}\{c = \tilde{\boldsymbol{y}}_{t,i}\}$, hence, even for above noisy target label case, the scores s_{ic_b} or s_{ic_u} will only increase to a certain absolute level regardless of other class score values, $\{s_{ic}\}_{c\in\mathcal{C}^{1:t-1}}$. Thus, at test time, a similar pixel still may be predicted as a past class in $\mathcal{C}^{t-1}$ since the frozen past classifiers would still output a considerable score — this subtle difference between the loss functions is illustrated in Figure 2.

Weight transfer from unknown class classifier Finally, for learning $\{\boldsymbol{\phi}^t_c\}_{c\in\mathcal{C}_t}$, we initialize all the filters with $\boldsymbol{\phi}^{t-1}_{c_u}$, the classifier learned for the unknown class c_u at the previous state. The reasoning is that the classes in $\mathcal{C}_t$ would have been labeled as c_u (as the potential future classes) in state $t-1$, hence, such weight transfer from $\boldsymbol{\phi}^{t-1}_{c_u}$ would yield stable and fast learning of $\{\boldsymbol{\phi}^t_c\}_{c\in\mathcal{C}_t}$.

3.5 Contribution 3: Tiny Exemplar Memory

Using exemplar-memory to store a small portion of past training data for CIL is backed with a theoretical finding [18] as well as strong empirical results [25, 28, 2, 1, 4] for classification.

Hence, we propose to use it for CISS as well with a tailored *class-balanced* sampling strategy. The main rationale of using the memory is to make sure to include at least one sample with correct GT label per each class in $\mathcal{C}^{1:t-1}$ in the training set for state t.

Namely, even though the pseudo-label $\hat{\boldsymbol{y}}_{t-1}$ can provide labels for the classes in $\mathcal{C}^{1:t-1}$, it is also possible that the given image $\boldsymbol{x}_t$ would never contain object cues for $\mathcal{C}^{1:t-1}$. In such case, it will be impossible to discriminate $\{c_b, c_u, \mathcal{C}_t\}$ from $\mathcal{C}^{1:t-1}$ and severe forgetting of $\mathcal{C}^{1:t-1}$ would occur. Therefore, by denoting $M = |\mathcal{M}|$, after learning incremental state $t-1$, we sample $M/|\mathcal{C}^{1:t-1}|$ data points from $\mathcal{D}_{t-1}$, and store them in the memory $\mathcal{M}$ by removing equal number of samples per each class in $\mathcal{C}^{1:t-2}$ from $\mathcal{M}$. In this way, $\mathcal{M}$ always contains at least one sample from each class in $\mathcal{C}^{1:t-1}$, and we show in the experiments that this class-balanced strategy is more helpful than random sampling [4] as is done for CIL for classification.

Algorithm 1: Pseudo code of SSUL-M

Require $f_{\boldsymbol{\theta}}$, $\mathcal{M}$, T, n_{epochs}, batch size K
$f_{\boldsymbol{\theta}} \leftarrow$ Initialize $h_{\boldsymbol{\psi}}$, $\boldsymbol{\phi}_{c_b}$, $\boldsymbol{\phi}_{c_u}$
Initialize $\mathcal{M}$
for $t \leftarrow 1, ..., T$ **do**
  $f^t_{\boldsymbol{\theta}} \leftarrow$ Initialize $\{\boldsymbol{\phi}^t_c\}_{c \in \mathcal{C}_t}$
  if $t \neq 1$ **then**
    `Freeze` $h_{\boldsymbol{\psi}}, \{\boldsymbol{\phi}^t_c\}_{c \in \mathcal{C}^{1:t-1}}$
    $\{\boldsymbol{\phi}^t_c\}_{c \in \mathcal{C}_t} \leftarrow$ `Weight Transfer`$(\boldsymbol{\phi}^{t-1}_{c_u})$
  end if
  for $n \leftarrow 1, ..., n_{\text{epochs}}$ **do**
    for $B_n \overset{K/2}{\sim} \mathcal{D}_t$ **do**
      $B_{\mathcal{M}} \overset{K/2}{\sim} \mathcal{M}$
      $\tilde{B}_n, \tilde{B}_{\mathcal{M}} \leftarrow$ `Label.Aug.`$(B_n, B_{\mathcal{M}})$
      $\boldsymbol{\theta} \leftarrow$SGD$(\tilde{B}_n \cup \tilde{B}_{\mathcal{M}}, \mathcal{L}^t_{\texttt{BCE}}(\boldsymbol{\theta}))$
    end for
  end for
  `Update` $\mathcal{M}$
end for
return $f^t_{\boldsymbol{\theta}}$

3.6 Summary

We summarize our SSUL-M algorithm in Algorithm 1, in which our contributions described in Section 3.3$\sim$ Section 3.5 are denoted in the typewriter font. Namely, `Label.Aug.` stands for generating $\tilde{\boldsymbol{y}}_t$ as in (1) for the selected batches $\mathcal{B}_n$ and $\mathcal{B}_{\mathcal{M}}$, "`Freeze`, $\mathcal{L}^t_{\texttt{BCE}}$, and `Weight Transfer`" denote the methods for the stable score learning described in Section 3.4, and `Update` $\mathcal{M}$ denote the exemplar-memory maintenance with the class balanced sampling as mentioned in Section 3.5. Note we are constructing a mini-batch by sampling equal amount of data from $\mathcal{D}_t$ and $\mathcal{M}$, hence, the samples in $\mathcal{M}$ act as anchor points to improve the precision of the predictions.

4 Experiments

4.1 Experimental Setting

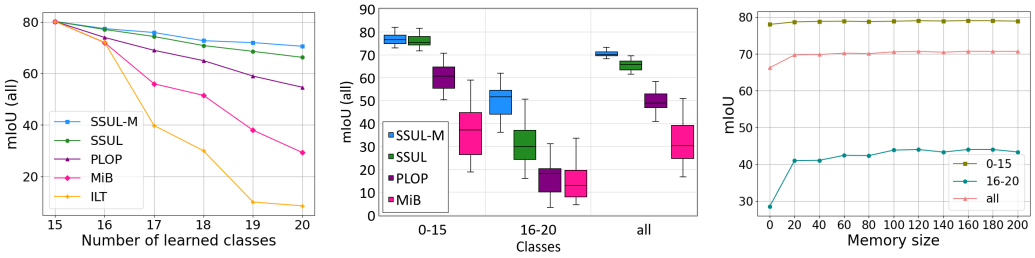
Dataset We followed the experimental setting of [3] and evaluated our method using Pascal-VOC 2012 [9] and ADE20K [30] datasets. Originally, [3] set two experimental protocols, *disjoint* and *overlapped*, but we believe the latter is more realistic and challenging. Therefore, we evaluated on the *overlapped* setup only. **Pascal VOC 2012** contains 20 foreground object classes and one background class, and **ADE20K** consists of 150 classes of both stuff and objects. We consider several incremental learning scenarios for each dataset, from the scenarios considered by the other baselines to newly proposed *challenging* scenarios with larger number of incremental states. A more detailed description on the datasets is introduced in the Supplementary Material (S.M.).

Evaluation Metrics We use the mean Intersection-over-Union (mIoU) as our evaluation metric, which computes the IoU for each class then computes the average over the classes. The IoU is defined as $IoU = \textit{true-positive}/(\textit{true-positive} + \textit{false-positive} + \textit{false-negative})$.

Implementation Details For all experiments, following other works [7, 3], we use a Deeplab-v3 segmentation network [5] with a ResNet-101 [10] backbone pre-trained on ImageNet [6]. We optimize the network using SGD with an initial learning rate of 10^{-2} and a momentum value of 0.9 for all CISS steps. Also, we set the learning rate schedule, data augmentation, and output stride of 16 following [5] for all experiments. For each incremental state t, we train the network for 50 epochs for

Table 1: Experimental results on Pascal VOC 2012.

Method	VOC 10-1 (11 tasks)			VOC 15-1 (6 tasks)			VOC 5-3 (6 tasks)			VOC 19-1 (2 tasks)			VOC 15-5 (2 tasks)		
	0-10	11-20	all	0-15	16-20	all	0-5	6-20	all	0-19	20	all	0-15	16-20	all
EWC [17]	-	-	-	0.30	4.30	1.30	-	-	-	26.90	14.00	26.30	24.30	35.50	27.10
LwF-MC [19]	4.65	5.90	4.95	6.40	8.40	6.90	20.91	36.67	24.66	64.40	13.30	61.90	58.10	35.00	52.30
ILT [23]	7.15	3.67	5.50	8.75	7.99	8.56	22.51	31.66	29.04	67.75	10.88	65.05	67.08	39.23	60.45
MiB [3]	12.25	13.09	12.65	34.22	13.50	29.29	57.10	42.56	46.71	71.43	23.59	69.15	76.37	49.97	70.08
PLOP [7]	44.03	15.51	30.45	65.12	21.11	54.64	17.48	19.16	18.68	75.35	**37.35**	73.54	75.73	**51.71**	70.09
SSUL	**73.78**	**41.13**	**58.23**	**78.06**	**28.54**	**66.27**	**71.17**	**45.38**	**52.75**	**78.52**	27.23	**76.07**	**77.42**	47.16	**70.21**
SSUL-M	**74.79**	**48.87**	**62.45**	**78.92**	**43.86**	**70.58**	**72.91**	**49.02**	**55.85**	**78.52**	**40.51**	**76.71**	**79.53**	**52.87**	**73.19**
Joint	78.41	76.35	77.43	79.77	72.35	77.43	76.91	77.63	77.43	77.51	77.04	77.43	79.77	72.35	77.43



(a) (b) (c)

Figure 3: (a): mIoU evaluation on VOC 15-1, (b): mIoU distributions for 20 different class-orderings for VOC 15-1, (c): mIOU on VOC 15-1 with varying memory size $|\mathcal{M}|$.

Pascal VOC with a batch-size of 32 and 60 epochs for ADE20K with a batch-size of 24. We tune the hyperparameters using 20% of the training set as a validation set and report the final results on the standard test set. For the exemplar memory, we utilized memory with a fixed size of $|\mathcal{M}| = 100$ for Pascal VOC and $|\mathcal{M}| = 300$ for ADE20K. To highlight the effect of the exemplar-memory, we report the results of the two versions of our method — SSUL (without memory) and SSUL-M (with memory). For the saliency-map detector to generate the Unknown class label, we employed DSS [11] pretrained on MSRA-B dataset [21], which contains 5,000 labels for salient objects. The experiments were implemented in PyTorch 1.7 with CUDA 10.1 using two NVIDIA V100 GPUs. More information on the experimental details are in S.M.

Baselines For a representative of the general regularization-based continual learning method, we select EWC [17] and LWC [19]-MC and applied them to each experimental setup of CISS. For CISS specific baselines, we compared with ILT [23], MiB [3] and PLOP [7], and the Joint Training (Joint) result is also given as an upper bound. Note that PLOP [7] is the current state-of-the-art. We reproduced the results of all baselines using the official code provided by the authors of [7].

4.2 Experimental results on benchmark dataset

Pascal VOC 2012 Following [7, 3], we evaluate our method on four different scenarios, *i.e.*, {10-1, 15-1, 15-5, and 19-1} as well as a more challenging scenario, {5-3}. The numbers in each scenario denote the number of classes to be trained for each state. For example, VOC 5-3 means learning 5 classes at the base task ($t = 1$), and then incrementally learning 3 classes five times (hence, $T = 6$).

In Table 1, we observe our SSUL consistently outperforms the baselines with *huge* margin in all scenarios, even without using the exemplar-memory. Furthermore, the gap widens particularly for more challenging scenarios, namely, for the cases in which the base task has fewer classes and the number of tasks is larger. Note although MiB [3] and PLOP [7] show robustness for simple 2 tasks scenarios (19-1 and 15-5), their performance are *rapidly* dropped in more challenging and practical scenarios (10-1, 5-3, 15-1). More specifically, Figure 3(a) shows the mIoU evolution for 15-1 scenario at each incremental step, and the baselines suffers from a significant drop of mIoU, as the new classes are incrementally learned. In contrast, SSUL improves mIoUs for both base (0-15) and newly learned (16-20) classes significantly, showing much slower drop of mIoU. These results shows that as long as the label augmentation of BG class with Unknown class and pseudo-labels is properly done, our stable score learning is very effective for CISS. Particularly, we observe that model freeze, which is believed to be not effective in CIL, is much more effective than the KD used in other baselines.

Table 2: Experimental results on ADE20K.

Method	ADE 100-5 (11 tasks)			ADE 100-10 (6 tasks)		
	0-100	101-150	all	0-100	101-150	all
ILT [23]	0.08	1.31	0.49	0.11	3.06	1.09
MiB [3]	36.01	5.66	25.96	38.21	11.12	29.24
PLOP [7]	39.11	7.81	28.75	40.48	13.61	31.59
SSUL	**42.03**	**15.80**	**33.35**	**42.10**	**16.02**	**33.46**
SSUL-M	**42.53**	**15.85**	**34.00**	**42.17**	**16.03**	**33.89**

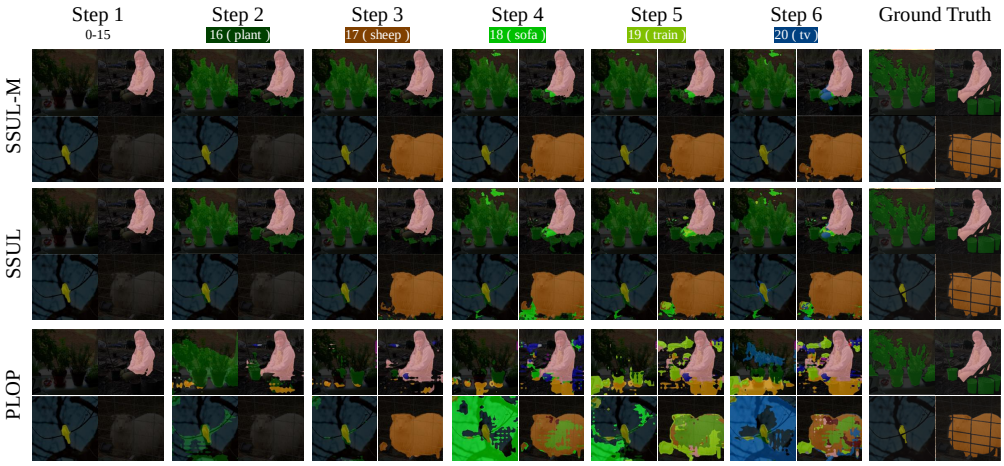


Figure 4: Qualitative results of SSUL-M, SSUL and PLOP on VOC 15-1 scenario.

Furthermore, we observe our SSUL-M, which uses exemplar-memory, further strengthens SSUL significantly, particularly for the newly learned classes (*i.e.*, for $t \geq 2$). For example, in VOC 15-1, by storing only $5 \sim 6$ images per class in $\mathcal{M}$, the mIoU for classes 16-20 improved about 15%. This confirms our intuition that the samples in the memory act as "anchors" to improve the precision of the predictions, hence, prevent forgetting. We can also clearly observe this improvement in Figure 4.

To verify the robustness of each method on various class orderings, we experimented on the 20 difference orderings on VOC 15-1 scenario, as proposed in [7]. Figure 3(b) plots the mIoU distributions for different methods, and we clearly observe both SSUL and SULL-M not only achieve higher mIoUs, but also have smaller variations compared to the baselines. In addition, we note SULL achieves $\times 1.5$ faster training time compared to PLOP due to network freezing.

Qualitative analysis In Figure 4, we visualized the qualitative results for four images from VOC 15-1 task. We observe that PLOP partly maintains knowledge learned from the base task (*bird* and *person*), however, it becomes fatal in forgetting the classes learned in the intermediate steps, such as plant and sheep. In addition, PLOP frequently produces many false-positive predictions, lowering the mIoU of several classes. (E.g., see *bird* in Step 5.) On the other hand, we observe SSUL and SSUL-M both maintains the previously learned classes with high stability and effectively learns new classes. For example, the *sheep* class is accurately learned at Step 3 and not forgotten afterwards. Moreover, we observe SULL-M achieved further improvement of both plasticity and stability over SULL, especially by reducing the false-positive predictions.

ADE20K Unlike VOC 2012, ADE20K is densely labeled for both stuff and thing with 150 classes. It means that the class definition in ADE20K is clearer therefore, it naturally make reduce the concern about the semantic drift of BG label. To make efficient use of this prior knowledge of dataset, we consider unlabeled pixels as unknown class without using the saliency detector and enlarge the size of memory to $|\mathcal{M}| = 300$. We evaluated our method in four different scenarios, $i.e.$, {100-5, 100-10, 50-50, 100-50} as in [7, 3]. Table 2 again shows that SSUL achieves superior performance in the more challenging tasks (100-5, 100-10) than other methods. We believe that, this result demonstrates SSUL is also quite effective in CISS for a densely labeled dataset, without any extra saliency detector. The qualitative analysis and quantitative comparison in other scenarios are provided in the S.M.

Table 3: Ablation study for SSUL on VOC 15-1 about PL (pseudo labeling), Unknown (unknown class modeling & weight transfer), Freeze (model freezing), and Sigmoid+BCE (stable score learning).

				15-1 (6 tasks)		
PL	Unknown	Freeze	Sigmoid+BCE	0-15	16-20	all
✓	✓	✓	✓	**78.06**	**28.54**	**66.27**
✓	✗	✓	✓	73.42	21.79	61.12
✓	✓	✗	✓	53.56	14.48	44.25
✓	✓	✓	✗	53.02	21.62	45.54

Table 4: Ablation study for saliency-map detector (left), weight transfer (mid), and memory sampling (right) on VOC 15-1.

Saliency-map detector				**Weight transfer**				**Memory sampling**			
Methods	0-15	16-20	all	Methods	0-15	16-20	all	Methods	0-15	16-20	all
DRFI [13]	77.46	27.21	65.63	*random* $\rightarrow \phi_c^t$	73.73	23.99	61.89	random	78.61	38.87	69.15
DSS [11]	78.06	28.54	66.27	$\phi_{c_b}^{t-1} \rightarrow \phi_c^t$	73.29	23.70	61.48	class-balanced	**78.92**	**43.86**	**70.58**
ground-truth	79.15	29.01	67.21	$\phi_{c_u}^{t-1} \rightarrow \phi_c^t$	**78.06**	**28.54**	**66.27**	-	-	-	-

4.3 Ablation Study

Ablation study on proposed components of SSUL Here, we analyze the effect of each proposed component of SSUL on VOC 15-1 scenario. Table 3 compares the mIoU of each ablation case, without memory, and the first low shows the result of SSUL with full components. Firstly, when the Unknown class is removed, we clearly observe that the mIoU's of both the first task (0-15) and subsequent classes (16-20) decrease. Hence, this clearly demonstrates the advantage of using the Unknown class label, in terms of increasing both the plasticity and stability. Secondly, note that model freezing has a significant impact on the performance of CISS. Specifically, it not only prevents catastrophic forgetting on the first task, but also plays a critical role to learn new classes well. Finally, we observe when Softmax+CE instead of Sigmoid+BCE is used (the last ablation), the forgetting of the first task (0-15) drops more significantly despite model freezing. We believe this confirms our intuition in Section 3.4 on why Sigmoid+BCE can lead to more stable score learning.

Saliency-map detector and weight transfer The first two columns in Table 4 shows the ablation study on saliency-map detector for Unknown class and weight transfer. We compare the result of a default neural network-based saliency map extractor, DSS [11] with ground-truth and a random forest-based DRFI [13]. We observe the differences of the mIoU's among them are quite small, therefore, we believe that the quality of saliency map is not a significant factor of our method. The ablation study on weight transfer demonstrates weights for new classes initialized by $\phi_{c_u}^{t-1}$ is most effective to learn it. It means, the representations of Unknown class label can be successfully transferred to learn new classes, even though hidden layers of the neural network are frozen.

Memory size and sampling rule The right column row of Table 4 shows the result of the mIoU on two sampling rules. We observe that, compared to random sampling, our proposed class-balanced sampling achieves better mIoU, particularly for the newly learned classes. We believe the reason is class-balanced sampling ensures at least one sample per class, therefore, it prevents the forgetting of minority classes than random sampling, which can miss certain classes in $\mathcal{M}$. Finally, Figure 3(c) shows the dependency of SSUL-M on the memory size. It illustrates that tiny exemplar-memory for CISS significantly helps to increase the mIoU for the newly learned classes (16-20) than the base classes (0-15). Moreover, we observe that after sufficiently large $|\mathcal{M}|$ the mIoU performance becomes robust.

5 Concluding Remarks

We proposed a new class-incremental learning method SSUL-M (Semantic Segmentation with Unknown Label with Memory) for semantic segmentation. In order to address two additional challenges of CISS, we made three main contributions — label augmentation with Unknown class labels, stable score learning, and tiny exemplar memory. They all were convincingly shown to be very effective in various CISS scenarios and our SSUL-M significantly outperformed baselines. We hope our analyses and insights can lead to new research directions for the CISS problem.

References

[1] Hongjoon Ahn, Jihwan Kwak, Subin Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. *arXiv preprint arXiv:2003.13947*, 2020.

[2] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018.

[3] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9233–9242, 2020.

[4] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

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

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. Plop: Learning without forgetting for continual semantic segmentation. *arXiv preprint arXiv:2011.11390*, 2020.

[8] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer vision-ECCV 2020-16th European conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XX*, volume 12365, pages 86–102. Springer, 2020.

[9] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip HS Torr. Deeply supervised salient object detection with short connections. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3203–3212, 2017.

[12] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019.

[13] Huaizu Jiang, Jingdong Wang, Zejian Yuan, Yang Wu, Nanning Zheng, and Shipeng Li. Salient object detection: A discriminative regional feature integration approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2083–2090, 2013.

[14] Beomyoung Kim, Sangeun Han, and Junmo Kim. Discriminative region suppression for weakly-supervised semantic segmentation. In *AAAI Conference on Artificial Intelligence*, 2021.

[15] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *International Conference on Machine Learning*, pages 5275–5285. PMLR, 2020.

[16] JangHyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. In *International Conference on Learning Representations*, 2021.

[17] James Kirkpatrick, Razvan Pascanu, Neil 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(13):3521–3526, 2017.

[18] Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is np-hard. In *International Conference on Machine Learning*, pages 5327–5337. PMLR, 2020.

[19] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[20] Jiang-Jiang Liu, Qibin Hou, Ming-Ming Cheng, Jiashi Feng, and Jianmin Jiang. A simple pooling-based design for real-time salient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3917–3926, 2019.

[21] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *IEEE Transactions on Pattern analysis and machine intelligence*, 33(2):353–367, 2010.

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

[23] Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[24] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[25] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2001–2010, 2017.

[26] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

[27] Yunchao Wei, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1568–1576, 2017.

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

[29] Qi Yao and Xiaojin Gong. Saliency guided self-attention network for weakly and semi-supervised semantic segmentation. *IEEE Access*, 8:14413–14423, 2020.

[30] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

6 Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The main claims of our paper are included in abstract and introduction section.
   (b) Did you describe the limitations of your work? [Yes] We describe the limitation of our work in concluding remarks section.
   (c) Did you discuss any potential negative societal impacts of your work? [No] There is no expected negative societal impacts
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code and instructions are in Supplement Materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All details for our method are described in the section of experimental settings.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We showed the experimental results on 20 different class orderings in the section of experiments
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] These are in experimental settings.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]