ALIFE: Adaptive Logit Regularizer and Feature Replay for Incremental Semantic Segmentation

Anonymous Author(s)
Affiliation
Address
email

Abstract

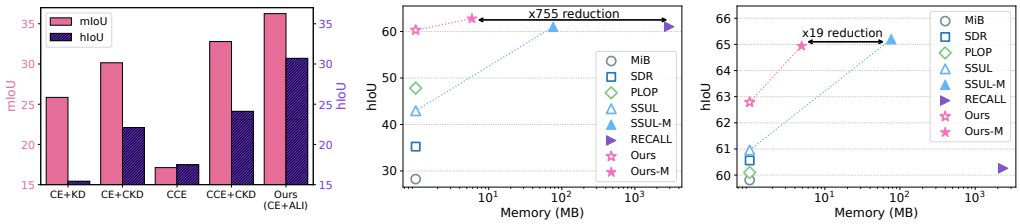
We address the problem of incremental semantic segmentation (ISS) recognizing novel object/stuff categories continually without forgetting previous ones that have been learned. The catastrophic forgetting problem is particularly severe in ISS, since pixel-level ground-truth labels are available only for the novel categories at training time. To address the problem, regularization-based methods exploit probability calibration techniques to learn semantic information from unlabeled pixels. While such techniques are effective, there is still lack of theoretical support. Replay-based methods propose to memorize a small set of images for previous categories. They achieve state-of-the-art performance at the cost of large memory footprint. We propose in this paper a novel ISS method, dubbed ALIFE, that provides a better compromise between accuracy and efficiency. To this end, we first show an in-depth analysis on the calibration techniques to better understand the effects on ISS. Based on this, we then introduce an adaptive logit regularizer (ALI) that enables our model to better learn new categories, while retaining knowledge for previous ones. We also present a feature replay scheme that memorizes features, instead of images directly, in order to reduce memory requirements significantly. Since a feature extractor is changed continually, memorized features should also be updated at every incremental stage. To handle this, we introduce category-specific rotation matrices updating the features for each category separately. We demonstrate the effectiveness of our approach with extensive experiments on standard ISS benchmarks, and show that our method achieves a better trade-off in terms of accuracy and efficiency. Our code and model will be made publicly available online.

1 Introduction

Humans are capable of learning new concepts continually, while preserving or even improving previously acquired knowledge. Artificial neural networks are, however, prone to forget the knowledge they have learned if being trained with samples for new object/scene categories alone. The reason for this problem, so-called catastrophic forgetting [11, 27], is that parameters of neural networks change abruptly to handle new categories without accessing training samples for previous categories. A straightforward way to alleviate the problem is to re-train a model with training examples for entire categories observed so far, which is however computationally demanding.

Incremental learning is an alternative approach to learning new categories continuously without re-training on the entire dataset. While many methods have been proposed for incremental classification [4, 21, 23, 25, 33, 36], a few attempts explore incremental semantic segmentation (ISS), where training images for new categories are partially labeled to reduce the cost for manual annotation. That is, pixels for new categories are labeled only, while remaining ones are marked as unknown. The unknown regions should be considered separately, since they could contain previous categories along with ones that would be seen in the future.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) IoU scores on ADE20K [39]. (b) hIoU-memory comparison on PASCAL VOC [10].

Figure 1: (a) Quantitative comparison of intersection-of-union (IoU) scores on ADE20K. Each model learns 50 novel categories after learning 100 categories. mIoU: an average IoU score for 150 categories. hIoU: a harmonic mean between two average IoU scores for previous (100) and novel (50) categories. (b) Quantitative comparison of hIoU and memory requirement on PASCAL VOC. (From left to right) Each method learns 1 and 5 novel categories after learning 20 and 16 categories, respectively. Ours-M memorizes 1K features for each previous category. For RECALL [26] and SSUL-M [3], we consider the size of memory required to store images only, and discard that for corresponding labels or saliency maps. Best viewed in color.

Current ISS methods can be categorized into regularization-based [2, 9, 30] and replay-based approaches [3, 26]. Regularization-based methods typically exploit knowledge distillation (KD) [15] to preserve the discriminative ability of ISS models for previous categories. In particular, the seminal work of [2] alleviates the semantic shift of unknown regions. Specifically, it introduces calibrated cross-entropy (CCE) and calibrated KD (CKD) terms to learn semantic information from unknown regions. Although CCE and CKD terms have shown the effectiveness on ISS methods [2, 9, 30], there is still lack of theoretical support. Replay-based methods propose to use a replay buffer consisting of web-crawled or previously seen images. The replay buffer provides rich information storing the knowledge for previous categories, but it incurs large memory footprint, particularly for the task of semantic segmentation that typically adopts high-resolution images.

In this paper, we present a novel ISS method, dubbed ALIFE, that 1) alleviates catastrophic forgetting and 2) reduces memory requirements. For the first aspect, we analyze gradients of CCE and CKD terms for better understanding the effectiveness on catastrophic forgetting, and make the following observations: (1) For unknown regions, CCE reduces logit values of new categories, which is crucial for preventing overfitting to the new categories. However, it always raises logit values of all previous categories, without considering whether predictions for those regions are correct or not, which lessens the discriminative power for previous categories. (2) CKD makes it difficult to distinguish new categories from a background one. Motivated by these observations, we introduce an adaptive logit regularizer (ALI) that enables better learning new categories and alleviating catastrophic forgetting for previous ones (Fig. 1(a)). For the second aspect, we propose to exploit latent features for replaying, reducing memory footprint and avoiding data privacy issues (*e.g.*, in medical imaging [35]). In contrast to the replay buffer storing images directly, our approach to using the features updates them at every incremental stage, as a feature extractor is also updated continually at training time. Specifically, we exploit category-specific rotation matrices using the Cayley transform. Rotating latent features is computationally efficient, while maintaining the relations between the features. We demonstrate that exploiting ALI with memorizing features achieves a better trade-off in terms of accuracy and efficiency (Fig. 1(b)). Our main contributions are summarized as follows:

- We show an in-depth analysis of probability calibration methods widely used for ISS [2, 9, 30], and introduce ALI that enables our model to better learn new categories, while maintaining the knowledge for previous categories.
- We present a novel replay strategy using category-specific rotation matrices, which helps to alleviate catastrophic forgetting for previous categories with much less memory requirements than replaying raw images.
- Extensive experiments demonstrate the effectiveness of our approach to using ALI and replaying features with rotation matrices. We set a new state of the art on standard ISS benchmarks [10, 39].

2 Related work

Many incremental learning approaches have been introduced trying to preserve knowledge for previous categories, while recognizing new ones [28]. They can be categorized into task-incremental

and class-incremental methods. Task-incremental methods [21, 23, 25] treat learning a new set of categories as a new task. With a perfect task identifier, they classify each task separately at test time. The perfect identifier is however often not available in practice [32]. On the contrary, class-incremental methods [4, 8, 33, 36] attempt to recognize all categories observed so far at once without the task identifier, being more practical. In the following, we describe class-incremental methods pertinent to ours.

Image classification. Incremental methods typically adopt a KD technique [15] in order to encourage a current model to imitate softmax probabilities [1, 33] or intermediate representations [8, 18] obtained from a previous one, retaining the discriminability for previous categories. Another line of works focuses on preventing a classifier from overfitting to new categories, by correcting biased weights of a classifier with a post-processing [36, 38] or normalizing a classifier [18]. Similarly, our ALI is beneficial to alleviating the overfitting problem and maintaining the discriminative power for previous categories, but differs in that it adaptively regularizes logit values during training. In addition, we exploit latent features for replaying, while all the aforementioned methods rely on an image replay strategy that entails practical issues such as data privacy and large memory requirements. To avoid these problems, recent works employ a pseudo replay scheme that synthesizes images [20, 31, 34] or latent features [37] of previous categories. However, they often require a generator [12], which should also be trained incrementally. Instead of using raw images or synthesizing features, the work of [19] proposes to store latent features. It trains a feature adaptation network (FAN) mapping the latent features into a new feature space. The FAN might be sub-optimal, since the same network applies to all features without differentiating categories. Adopting individual FANs for each category might address the problem, but this requires high computational cost and large memory footprint. Differently, we propose to train category-specific rotation matrices. This is more efficient and accurate, as a rotation matrix is light-weight and features of each category are updated separately.

Semantic segmentation. A relatively few methods address the task of ISS, which can further be classified into regularization-based and replay-based approaches. The first approaches [2, 9, 30] focus on preserving knowledge for previous categories without using an experience replay. For example, MiB [2] proposes to consider the semantic shift of a background category in ISS, and introduces CCE and CKD terms using probability calibration techniques. While the calibration techniques are intuitive and effective, there are still lack of theoretical explanations. We provide a gradient analysis on CCE and CKD, and point out that CCE and CKD could disturb preserving knowledge for previous categories and discriminating new categories from the background category, respectively. SDR [30] and PLOP [9] propose to further regularize a current model in a latent feature space. Specifically, SDR minimizes distances between features of the same category [6], while PLOP distills intermediate features using a multi-scale strip pooling technique [8, 17]. The second approaches [3, 26] rely on an image replay strategy. RECALL [26] exploits web-crawled images with pseudo labels at each incremental stage, while SSUL [3] maintains a small set of images for previous categories, together with corresponding ground-truth masks. Both methods show state-of-the-art results at the cost of large memory footprint. Our approach differs in two aspects: (1) We exploit latent features for replaying, which reduces the size of required memory significantly; (2) Our method updates a feature extractor continually, and thus it is more flexible than RECALL and SSUL freezing the extractor.

3 Method

In this section, we first describe the ISS task briefly (Sec. 3.1), and then present our approach consisting of two steps at every incremental stage. Specifically, we introduce ALI based on an in-depth analysis for CCE and CKD (Sec. 3.2), and present a feature replay scheme using category-specific rotation matrices (Sec.3.3). Please refer to the supplement for more analysis and detailed derivations.

3.1 Problem statement

Following the common practice [2, 3, 9, 26, 30], we consider a series of T learning stages. Each stage t has its own dataset, $D^t = \{(x_i, y_i)\}_{i=1}^{n^t}$ of size n^t, where x and y are an image and a corresponding ground-truth mask, respectively. For incremental stages (*i.e.*, $t > 1$), we have two disjoint sets, C^t_{prev} and C^t_{new}, for previously learned and novel categories, respectively. Note that

ground-truth masks are labeled only for the categories of C^t_{new}. Formally, we denote by $y(\mathbf{p})$ a ground-truth label at position $\mathbf{p}$, and define labeled regions as follows:

$$\mathcal{R}^t_{\text{new}} = \bigcup_{c \in C^t_{\text{new}}} \mathcal{R}_c, \tag{1}$$

where $\mathcal{R}_c$ is a set of locations labeled as a category c, that is, $\mathcal{R}_c = \{\mathbf{p} \mid y(\mathbf{p}) = c\}$.

Our goal is to train a model that recognizes all categories observed so far, C^t_{all}, the union of the disjoint sets, *i.e.*, $C^t_{\text{all}} = C^t_{\text{prev}} \cup C^t_{\text{new}}$. In detail, our model consists of a feature extractor ϕ and a classifier w. The feature extractor takes an image and outputs a convolutional feature map for prediction, $\phi^t : x \mapsto f^t$, where we denote by $f^t(\mathbf{p}) \in \mathbb{R}^D$ a D-dimensional feature vector at position $\mathbf{p}$. A logit value for a category c is then computed by the dot product between a feature and a classifier weight for the category c, $w^t_c \in \mathbb{R}^D$, as follows:

$$z^t_c(\mathbf{p}) = w^t_c \cdot f^t(\mathbf{p}), \tag{2}$$

where we omit bias terms of the classifier for brevity.

3.2 Step 1

In the first step, we initialize network weights of a model at the stage t with those for the previous stage $t-1$, and train a feature extractor and a classifier of the current model with a corresponding dataset D^t. A simple way to mitigate catastrophic forgetting for ISS is employing cross-entropy (CE) and KD [15] terms. To this end, we compute probabilities for a category c at position $\mathbf{p}$ as follows:

$$p^t_c(\mathbf{p}) = \frac{e^{z^t_c(\mathbf{p})}}{\sum_{k \in C^t_{\text{all}}} e^{z^t_k(\mathbf{p})}}, \quad c \in C^t_{\text{all}} \quad \text{and} \quad q^t_c(\mathbf{p}) = \frac{e^{z^t_c(\mathbf{p})}}{\sum_{k \in C^t_{\text{prev}}} e^{z^t_k(\mathbf{p})}}, \quad c \in C^t_{\text{prev}}, \tag{3}$$

which are obtained by applying the softmax function to logit values across C^t_{all} and C^t_{prev}, respectively. The CE and KD terms for ISS are then defined as follows:

$$L_{\text{CE}}(\mathbf{p}) = -\log p^t_{c^*}(\mathbf{p}), \quad \mathbf{p} \in \mathcal{R}^t_{\text{new}} \quad \text{and} \quad L_{\text{KD}}(\mathbf{p}) = \sum_{k \in C^t_{\text{prev}}} -p^{t-1}_k(\mathbf{p}) \log q^t_k(\mathbf{p}), \quad \forall \mathbf{p}, \tag{4}$$

where $c^* = y(\mathbf{p})$. The CE term is defined for the labeled regions $\mathcal{R}_{\text{new}}$ only, indicating that logit values of previous categories always decrease. Exploiting the CE term alone is thus prone to overfitting to new categories and leads to catastrophic forgetting. The KD term addresses this problem by transferring the discriminative power of a previous model, trained to classify previous categories, into the current one. Note that $C^t_{\text{prev}} = C^{t-1}_{\text{all}}$, and the KD term computes softmax probabilities across previous categories without considering new ones. Instead of exploiting CE and KD terms, the seminal work [2] proposes to use CCE and CKD, widely adopted in ISS [2, 9, 30]. In the following, we analyze gradients of CCE and CKD to better understand the influences on ISS, and introduce ALI to train our model.

CCE. The CCE loss [2] also computes a probability for unlabeled regions by summing the probabilities over all previous categories as follows:

$$L_{\text{CCE}}(\mathbf{p}) = \begin{cases} -\log p^t_{c^*}(\mathbf{p}), & \mathbf{p} \in \mathcal{R}^t_{\text{new}} \\ -\log p^t_{\text{cce}}(\mathbf{p}), & \mathbf{p} \notin \mathcal{R}^t_{\text{new}} \end{cases}, \tag{5}$$

where $p^t_{\text{cce}}(\mathbf{p}) = \sum_{k \in C^t_{\text{prev}}} p^t_k(\mathbf{p})$. This marginal difference from the vanilla CE term brings a significant improvement on ISS. To analyze the reason behind the improvement, we summarize in Table 1 gradients of L_{CCE} w.r.t z^t_c. The first two rows show that CCE updates logit values for the labeled regions in the same way as in L_{CE}. We can see from last two rows that CCE also provides gradients for the unlabeled regions. Specifically, the third row shows that CCE reduces a logit value for new categories by the corresponding probability p^t_c, if $\mathbf{p} \notin \mathcal{R}^t_{\text{new}}$. This is reasonable, since all features on those regions do not at least belong to new categories. CCE thus helps to avoid the overfitting problem, alleviating catastrophic forgetting. Considering the fact that q^t_c is always larger than p^t_c (See the supplement), we can see from the last row that CCE raises

Table 1: Gradients of CCE w.r.t z^t_c.

Conditions		Gradients
$\mathbf{p} \in \mathcal{R}^t_{\text{new}}$	$c = y(\mathbf{p})$	$p^t_c - 1$
	$c \neq y(\mathbf{p})$	p^t_c
$\mathbf{p} \notin \mathcal{R}^t_{\text{new}}$	$c \in C^t_{\text{new}}$	p^t_c
	$c \in C^t_{\text{prev}}$	$p^t_c - q^t_c$

logit values for all previous categories by $q_c^t - p_c^t$ in the unlabeled regions. This prevents the logit values of previous categories from continuing to decrease in the unlabeled regions, which is however effective only when predictions of the current model are correct. Otherwise, when predictions are incorrect, CCE rather raises logit values of the wrongly predicted categories, aggravating the incorrect predictions.

CKD. Assuming that categories of C_{new}^t are likely to be labeled as a background one in the previous stage, the CKD loss [2] is defined as follows:

$$L_{\text{CKD}}(\mathbf{p}) = -p_{bg}^{t-1}(\mathbf{p}) \log p_{\text{ckd}}^t(\mathbf{p}) + \sum_{k \in C_{\text{prev}}^t \setminus \{bg\}} -p_k^{t-1}(\mathbf{p}) \log p_k^t(\mathbf{p}), \quad \forall \mathbf{p}, \tag{6}$$

where $p_{\text{ckd}}^t(\mathbf{p}) = \sum_{k \in \{bg\} \cup C_{\text{new}}^t} p_k^t(\mathbf{p})$. For previous categories except the background, this term enables transferring knowledge from p_k^{t-1} to p_k^t directly. Similarly, a vanilla KD term in Eq. (4) also encourages a current model to output probabilities similar to p_k^{t-1} for $k \in C_{\text{prev}}^t$, but it distills the knowledge from p_k^{t-1} into q_k^t without considering logit values for new categories. To compare KD and CKD, we compute gradients w.r.t z_c^t for $c \in C_{\text{prev}}^t \setminus \{bg\}$ as follows:

$$\frac{\partial L_{\text{KD}}(\mathbf{p})}{\partial z_c^t(\mathbf{p})} = q_c^t(\mathbf{p}) - p_c^{t-1}(\mathbf{p}) \quad \text{and} \quad \frac{\partial L_{\text{CKD}}(\mathbf{p})}{\partial z_c^t(\mathbf{p})} = p_c^t(\mathbf{p}) - p_c^{t-1}(\mathbf{p}). \tag{7}$$

Here we provide two cases to explain the reason why CKD better transfers the knowledge for previous categories except the background: (1) When q_c^t and p_c^{t-1} are equal, the gradient of KD w.r.t z_c^t is zero, indicating that z_c^t remains the same. However, note that p_c^t is lower than p_c^{t-1} in this case, since p_c^t is always lower than q_c^t. This suggests that z_c^t should rather increase in order that p_c^t follows the target probability p_c^{t-1}. The gradient of CKD w.r.t z_c^t has a negative value in the same case (*i.e.*, $q_c^t = p_c^{t-1}$), suggesting that z_c^t increases by gradient descent. (2) When the current model already provides the same probability as the target one (*i.e.*, $p_c^t = p_c^{t-1}$), it is unnecessary to adjust z_c^t. The gradient of KD w.r.t z_c^t has a positive value, as q_c^t is larger than p_c^t (or equivalently p_c^{t-1} in this case). This indicates that z_c^t decreases by gradient descent, and p_c^t in turn becomes lower than p_c^{t-1}. On the other hand, CKD maintains z_c^t the same, since its gradient w.r.t z_c^t is zero. These examples describe the effectiveness of CKD. On the contrary, CKD also has negative effects on ISS. Let us suppose the gradients of CKD w.r.t z_c^t for $c \in \{bg\} \cup C_{\text{new}}^t$ as follows:

$$\frac{\partial L_{\text{CKD}}(\mathbf{p})}{\partial z_c^t(\mathbf{p})} = \left(p_{\text{ckd}}^t(\mathbf{p}) - p_{bg}^{t-1}(1\mathbf{p}) \right) \frac{p_c^t(\mathbf{p})}{p_{\text{ckd}}^t(\mathbf{p})}. \tag{8}$$

The gradient of CKD in Eq. (8) has a negative value, when p_{ckd}^t is lower than p_{bg}^{t-1}. Thus, logit values for background and new categories increase by gradient descent. This in turn raises probabilities for background and new categories all together, disturbing discriminating the new categories from the background at training time. In case of $p_{\text{ckd}}^t > p_{bg}^{t-1}$, CKD reduces the logit values for background and new categories, which is however problematic when $p_{bg}^t < p_{bg}^{t-1}$. In this case, the logit value for the background category z_{bg}^t should increase.

ALI. We have shown that CCE alleviates catastrophic forgetting, while CKD better guides transferring the knowledge of a previous model than KD. To avoid the negative effects of CCE and CKD, we present in Table 2 a new form of gradients w.r.t z_c^t. The first row is identical to the third one of Table 1 that alleviates catastrophic forgetting. The second row is similar to the gradients of CKD in Eq. (7) that better capture knowledge for previous categories. Note that CKD excludes the background category in Eq. (7), since it assumes that new categories of the current stage (C_{new}^t) are marked as the background one at the previous stage. Differently, ours enables computing gradients for all previous categories (See the second row), since it does not require the assumption. Accordingly, this form of gradients incorporates the advantages of CCE and CKD, while discarding the negative effects. Using the gradients in Table 2, we define ALI as follows:

Table 2: Gradients of ALI w.r.t z_c^t.

Conditions		Gradients
$\mathbf{p} \notin \mathcal{R}_{\text{new}}^t$	$c \in C_{\text{new}}^t$	p_c^t
	$c \in C_{\text{prev}}^t$	$p_c^t - p_c^{t-1}$

$$L_{\text{ALI}}(\mathbf{p}) = \log \left(\sum_{k \in C_{\text{all}}^t} e^{z_k^t(\mathbf{p})} \right) - \sum_{k \in C_{\text{prev}}^t} p_k^{t-1}(\mathbf{p}) z_k^t(\mathbf{p}), \quad \mathbf{p} \notin \mathcal{R}_{\text{new}}^t. \tag{9}$$

The first term applies the log-sum-exp function to logit values across C^t_{all}, approximating the maximum logit value over C^t_{all}. The second term computes a weighted average of logit values for C^t_{prev} with probabilities of a previous model. In this context, ALI can be viewed as minimizing the difference between the maximum logit value and the weighted average adaptively. That is, it reduces the maximum logit value, while raising the logit values for previous categories. Note that ALI does not require computing either p^t_{cce} in CCE or p^t_{ckd} in CKD [2].

Training. To train our model, we use CE and ALI terms for labeled and unlabeled regions, respectively. We also apply a vanilla KD term for labeled regions to further regularize our model. An overall objective for the first step is defined as follows:

$$L_{\text{S1}}(\mathbf{p}) = L_{\text{CE}}(\mathbf{p}) + \lambda_{\text{ALI}} L_{\text{ALI}}(\mathbf{p}) + \lambda_{\text{KD}} L_{\text{KD}}(\mathbf{p}) \mathbb{1}[\mathbf{p} \in \mathcal{R}^t_{\text{new}}], \tag{10}$$

whereλ_{ALI} and λ_{KD} are balance parameters. We denote by $\mathbb{1}[\cdot]$ an indicator function whose output is 1 if the argument is true, and 0 otherwise.

Memorizing features. After training our model, we extract features for prediction and store them to replay for the next stage. Specifically, given input images containing one of new categories, a feature extractor first produces feature maps. For each image, features for each category are then averaged using a ground-truth mask. This process is repeated until the number of features for each category reaches a preset number S. That is, we memorize S features for each category. See the supplement for detailed descriptions.

3.3 Step 2

After the first step, we freeze a feature extractor ϕ^t and fine-tune a classifier w^t with a current dataset D^t along with memorized features extracted from the previous stage $t-1$. The memorized features, however, do not lie in the current feature space. To address this, we exploit category-specific rotation matrices to update features of each category separately, and present two loss terms to train the matrices. In the following, we provide detailed descriptions for training matrices and fine-tuning classifiers.

Training matrices. Memorized features, extracted in the previous stage $t-1$, are not compatible with a current classifier w^t. To address this, the work of [19] in image classification proposes to exploit two-layer perceptrons, called FAN, where the number of parameters is roughly $32D^2$. Two main limitations of FAN are as follows: (1) It updates all features without considering categories. Adopting separate FANs for each category alleviates this issue, but it is computationally expensive. (2) FAN ignores the relations between memorized features. The structural information in the feature space is crucial for the generalization ability of classifiers. To address these problems, we propose to train category-specific rotation matrices to update features of each category separately. The rotation transform enables maintaining the relations between features within the same category, while the number of parameters for each matrix is $0.5(D^2 - D)$ only (See the supplement). To this end, we first define a skew-symmetric matrix $\mathbf{S}_c$ of size $D \times D$ for the category $c \in C^t_{\text{prev}}$. A rotation matrix for the category c is then defined using the Cayley transform as follows:

$$\mathbf{R}_c = (\mathbf{I} - \mathbf{S}_c)(\mathbf{I} + \mathbf{S}_c)^{-1}, \tag{11}$$

where $\mathbf{I}$ indicates an identity matrix. Learning the parameters for the matrices is challenging, since training samples of D^t are labeled only for C^t_{new}. To handle this, we extract features f^{t-1} from a previous feature extractor ϕ^{t-1}, and compute a correlation score as follows:

$$s_c(\mathbf{p}) = \sum_{s=1}^{S} \text{ReLU}\left(\frac{f^{t-1}(\mathbf{p})}{\|f^{t-1}(\mathbf{p})\|} \cdot \frac{m_c(s)}{\|m_c(s)\|} \right), \tag{12}$$

where we denote by $m_c(s)$ the s-th item in memorized features of the category c. This allows to identify features associated with previous categories C^t_{prev}, since f^{t-1} and $m_c(s)$ share the same feature space. We then apply the softmax function over the correlation score as follows:

$$\sigma_c(\mathbf{p}) = \frac{e^{\tau s_c(\mathbf{p})}}{\sum_{\mathbf{p}} e^{\tau s_c(\mathbf{p})}}, \tag{13}$$

where τ is a temperature controlling the sharpness of σ_c. Note that both features f^{t-1} and f^t at position $\mathbf{p}$ encode the same semantic information. Using this fact, we define prototypes of the category c for previous and current stages, r_c^{t-1} and r_c^t, respectively, by computing a weighted average of each feature map, as follows:

$$r_c^{t-1} = \sum_{\mathbf{p}} \sigma_c(\mathbf{p}) f^{t-1}(\mathbf{p}), \quad r_c^t = \sum_{\mathbf{p}} \sigma_c(\mathbf{p}) f^t(\mathbf{p}). \tag{14}$$

We can match the prototypes for each category c via the corresponding rotation matrix $\mathbf{R}_c$. Namely, each matrix $\mathbf{R}_c$ rotates a previous prototype r_c^{t-1} to align it with a current one r_c^t. To train the matrix $\mathbf{R}_c$, we define an objective function as follows:

$$L_{\mathrm{ROT}} = \lambda_{\mathrm{FID}} L_{\mathrm{FID}} + L_{\mathrm{REG}}, \tag{15}$$

where we denote by L_{FID} and L_{REG} fidelity and regularization terms, respectively, balanced by the parameter λ_{FID}. The fidelity term maximizes cosine similarity as follows:

$$L_{\mathrm{FID}} = \sum_{c \in C_{\mathrm{prev}}^t} \left(1 - \frac{\hat{r}_c}{\|\hat{r}_c\|} \cdot \frac{r_c^t}{\|r_c^t\|} \right), \tag{16}$$

where $\hat{r}_c = \mathbf{R}_c r_c^{t-1}$. This encourages the matrices $\mathbf{R}_c$ to align $\hat{r}_c$ with r_c^t. The regularization term enforces the rotated prototypes $\hat{r}_c$ to be compatible with the current classifier w^t to better guide the alignment process. To this end, we compute a CE loss using the softmax classifier w^t as follows:

$$L_{\mathrm{REG}} = \sum_{c \in C_{\mathrm{prev}}^t} -\log \left(\frac{e^{\hat{r}_c \cdot w_c^t}}{\sum_{i \in C_{\mathrm{all}}^t} e^{\hat{r}_c \cdot w_i^t}} \right). \tag{17}$$

Note that we freeze both a feature extractor and a classifier when training rotation matrices.

Fine-tuning. After training rotation matrices, we update memorized features of each category as follows:

$$\hat{m}_c(s) = \mathbf{R}_c m_c(s). \tag{18}$$

The updated features along with training samples of D^t are then used to fine-tune a classifier w^t with the following objective:

$$L_{\mathrm{S2}}(\mathbf{p}) = L_{\mathrm{FL}}(\mathbf{p}) + \lambda_{\mathrm{ALI}} L_{\mathrm{ALI}}(\mathbf{p}) + \lambda_{\mathrm{MEM}} L_{\mathrm{MEM}}. \tag{19}$$

We denote by L_{FL} and L_{MEM} focal loss (FL) [24] and CE terms, respectively, defined as follows:

$$L_{\mathrm{FL}}(\mathbf{p}) = -(1 - p_{\hat{c}}^t(\mathbf{p}))^{\alpha} \log p_{\hat{c}}^t(\mathbf{p}), \quad \hat{c} = \begin{cases} y(\mathbf{p}) & , \quad \mathbf{p} \in \mathcal{R}_{\mathrm{new}}^t \\ \mathrm{argmax}_{k \in C_{\mathrm{prev}}^t} p_k^{t-1}(\mathbf{p}), & \mathbf{p} \notin \mathcal{R}_{\mathrm{new}}^t \end{cases}, \tag{20}$$

and

$$L_{\mathrm{MEM}} = \sum_{c \in C_{\mathrm{prev}}^t} \sum_{s=1}^{S} -\log \left(\frac{e^{\hat{m}_c(s) \cdot w_c^t}}{\sum_{k \in C_{\mathrm{all}}^t} e^{\hat{m}_k(s) \cdot w_k^t}} \right), \tag{21}$$

where we set α to 2 by default. Following [3, 9, 26, 30], we mark unlabeled regions in the training samples of D^t as predictions obtained from a previous model on-the-fly.

4 Experiments

In this section, we present a quantitative comparison between our method and the state of the art, and show ablation studies. More results including qualitative comparisons can be found in the supplement.

4.1 Implementation details

Dataset and evaluation. We evaluate our method on standard ISS benchmarks (PASCAL VOC [10] and ADE20K [39]). PASCAL VOC provides $10,582$ training [13] and $1,449$ validation samples with 20 object and one background categories, while ADE20K consists of $20,210$ and $2,000$ samples for training and validation, respectively, with 150 object/stuff categories. There are three incremental

Table 3: Quantitative results on ADE20K [39] in terms of IoU scores. SSUL-M [3] uses a replay buffer that consists of 300 previously seen images together with corresponding ground-truth labels. Numbers in bold are the best performance, while underlined ones are the second best. We show standard deviations in parentheses. †: Results are obtained by running the source codes provided the authors.

Methods	100-50(1)				50-100(2)				100-50(5)			
	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU
ILT [29]	18.29	14.40	17.00	16.11	3.53	12.85	9.70	5.54	0.08	1.31	0.49	0.15
MiB [2]	40.52	17.17	32.79	24.12	45.57	21.01	29.31	28.76	38.21	11.12	29.24	17.23
PLOP† [9]	42.10 (0.02)	16.22 (0.15)	33.53 (0.06)	23.42 (0.15)	48.24 (0.03)	21.31 (0.08)	30.40 (0.06)	29.56 (0.08)	40.78 (0.04)	14.02 (0.03)	31.92 (0.04)	20.87 (0.03)
SSUL [3]	41.28	18.02	33.58	25.09	**48.38**	20.15	29.56	28.45	40.20	18.75	33.10	25.57
Ours-S1	41.95 (0.05)	24.87 (0.27)	36.29 (0.06)	31.22 (0.20)	48.28 (0.15)	28.23 (0.17)	35.00 (0.14)	35.63 (0.15)	40.87 (0.08)	21.18 (0.14)	34.35 (0.07)	27.90 (0.12)
Ours-S2	**42.19** (0.08)	**25.07** (0.33)	**36.52** (0.07)	**31.45** (0.24)	48.29 (0.15)	**28.42** (0.16)	**35.13** (0.14)	**35.78** (0.16)	**41.05** (0.09)	**21.73** (0.18)	**34.65** (0.03)	**28.42** (0.14)
SSUL-M [3]	42.79	17.54	34.37	24.88	49.12	20.10	29.77	28.53	42.86	17.66	34.46	25.01

scenarios for each dataset, where we denote by each scenario A-$B(C)$. A, B and C indicate the number of categories at a base stage, the total number of novel categories, and the number of incremental stages, respectively. We follow the same scenarios as in [2, 3, 9, 26, 30], unless otherwise specified. Following the common practice [3, 9], we focus on an overlapped setting, where unlabeled regions could contain either previous or future categories. For evaluation, we report IoU scores on the validation set for each dataset, and do not exploit test-time augmentation or dense CRF techniques [22]. We denote by $\text{mIoU}_{\text{base}}$ and mIoU_{new} the mean IoU (mIoU) scores over base and new categories, respectively. Computing an mIoU score over all categories is typical, but we have found that this does not reflect IoU scores for new categories well, and those for base categories mostly influence evaluation results. To address this, we propose to use the harmonic mean (hIoU) of $\text{mIoU}_{\text{base}}$ and mIoU_{new}. For all experiments, we report scores averaged over 3 runs.

Training. Following [2, 3, 9, 26, 30], we adopt DeepLab-V3 [5] with ResNet-101 [14]. ResNet-101 is initialized with pre-trained weights for classification on ImageNet [7]. We use the SGD optimizer with an initial learning rate set to 1e-2 and 1e-3 for base and incremental stages, respectively. DeepLab-V3 is trained for 30 and 60 epochs at a base stage ($t = 1$) on PASCAL VOC [10] and ADE20K [39], respectively. For each incremental stage ($t > 1$), we perform a cross-validation to choose the number of epochs on PASCAL VOC, while fixing it to 60 on ADE20K. We train rotation matrices for 10 epochs using the SGD optimizer with an initial learning rate of 1e-2, and fix a preset number S and a temperature value τ to $1,000$ and 10 for all experiments. We fine-tune a classifier for 1 epoch using the SGD optimizer with an initial learning rate of 1e-3. For all experiments, we adjust the learning rate by the poly schedule. We provide a detailed description of hyperparameter settings in the supplement.

4.2 Comparison with the state of the art

ADE20K. We compare in Table 3 our approach with state-of-the-art methods, including MiB [2], PLOP [9], and SSUL [3]. Note that RECALL [26] is not designed to handle stuff categories, and results on ADE20K [39] are not available. From this table, we have three findings as follows: (1) Our approach exploiting the first step only, denoted by *Ours-S1*, already outperforms all other methods in terms of both mIoU and hIoU scores by significant margins for all scenarios. This validates the effectiveness of our approach without memorizing features. In particular, we can see that *Ours-S1* even outperforms SSUL-M [3] that memorizes 300 images along with ground-truth labels for replaying. A plausible reason is that SSUL freezes a feature extractor, limiting the flexibility to deal with new categories. (2) *Ours-S1* shows substantial IoU gains over MiB [2] using CCE and CKD for all scenarios. This verifies that both CCE and CKD are not always helpful for ISS. ALI is free from the limitations of CCE and CKD, and it allows our model to better learn new categories without forgetting previous ones. (3) Our approach memorizing features, denoted by *Ours-S2*, improves the performance over *Ours-S1* in terms of all metrics for all scenarios. Note that SSUL-M even performs worse than SSUL without replaying for 100-50(1) and 100-50(6) cases. Considering that we rely on at least 58 times less memory requirements than SSUL-M on ADE20K, the gains from memorizing features are remarkable compared to those of SSUL-M over SSUL.

Table 4: Quantitative results on PASCAL VOC [10] in terms of IoU scores. SSUL-M [3] memorizes 100 images in total, while RECALL [26] uses 500 images for each previous category. Note that both SSUL and SSUL-M also require an off-the-shelf saliency detector [16] on PASCAL VOC. Numbers in bold are the best performance, while underlined ones are the second best. We show standard deviations in parentheses. †: Results are obtained with the source codes provided the authors.

Methods	20-1(1)				16-5(1)				16-5(5)			
	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU
EWC [21]	26.90	14.00	26.30	18.42	24.30	35.50	27.10	28.85	0.30	4.30	1.30	0.56
LwF-MC [23]	64.40	13.30	61.90	22.05	58.10	35.00	52.30	43.68	6.40	8.40	6.90	7.26
ILT [29]	67.75	10.88	65.05	18.75	67.08	39.23	60.45	49.51	8.75	7.99	8.56	8.35
MiB† [2]	70.42 (0.13)	17.70 (1.89)	67.91 (0.17)	28.25 (2.44)	76.68 (0.11)	49.03 (0.27)	70.09 (0.14)	59.81 (0.23)	37.98 (0.72)	12.28 (0.19)	31.86 (0.57)	18.56 (0.28)
SDR [30]	71.30	23.40	69.00	35.24	76.30	50.20	70.10	60.56	47.30	14.70	39.50	22.43
PLOP† [9]	75.89 (0.21)	34.90 (0.93)	73.94 (0.24)	47.81 (0.91)	76.37 (0.14)	49.55 (0.29)	69.98 (0.18)	60.10 (0.26)	64.51 (0.13)	19.93 (0.20)	53.90 (0.06)	30.45 (0.21)
Ours-S1	75.90 (0.40)	50.04 (0.53)	74.67 (0.36)	60.31 (0.29)	76.87 (0.23)	53.07 (0.06)	71.21 (0.16)	62.79 (0.07)	63.24 (0.60)	21.86 (0.90)	53.39 (0.67)	32.49 (1.07)
Ours-S2	**76.25** (0.34)	**53.35** (0.63)	**75.16** (0.30)	**62.77** (0.34)	**78.04** (0.04)	**55.34** (0.30)	**72.63** (0.05)	**64.76** (0.19)	**66.46** (0.58)	**27.71** (1.43)	**57.24** (0.75)	**39.10** (1.51)
SSUL [3]	77.73	29.68	75.44	42.96	77.82	50.10	71.22	60.96	77.31	36.59	67.61	49.67
SSUL-M [3]	**78.83**	49.76	**76.49**	61.01	**78.40**	**55.80**	**73.02**	**65.20**	**78.36**	49.01	**71.37**	**60.30**
RECALL [26]	68.10	**55.30**	68.60	**61.04**	67.70	54.30	65.60	60.26	67.80	**50.90**	64.80	58.15

PASCAL VOC. We show in Table 4 quantitative results on PASCAL VOC [10]. Note that a comparison of SSUL [3] (SSUL-M) and other methods including ours is not fair, since it additionally exploits an off-the-shelf saliency detector [16] that brings significant improvements on PASCAL VOC. From this table, we can see that *Ours-S1* outperforms other methods in terms of hIoU scores for all scenarios, further demonstrating the effectiveness of our approach without memorizing features. We can also see that *Ours-S2* gives substantial gains over *Ours-S1* in terms of all metrics for all scenarios. In particular, the largest mIoU and hIoU gains of 3.85% and 6.61%, respectively, are reported on 16-5(5). This confirms once again that our feature replay scheme is effective even for the most challenging scenario on PASCAL VOC. SSUL-M [3] and RECALL [26] largely outperform ours only for 16-5(5), but they require at least 604 and 19 times more memory footprint than *Ours-S2*, respectively. Moreover, RECALL has difficulty for handling stuff categories, and SSUL performs poorly on ADE20K [39], where the saliency detector is not applicable. Our approach is versatile in that it is free to handle stuff categories without performance degradation on ADE20K (See Table 3).

4.3 Ablation study

We show in Table 5(a) an ablation analysis on different loss terms of our approach without memorizing features (Eq. (10)). From this table, we can see that using the CE term alone performs poorly due to catastrophic forgetting. We can also see that ALI alleviates the catastrophic forgetting problem remarkably in the second row. The last row shows that applying the KD term for labeled regions further gives IoU gains. We report in Table 5(b) IoU scores using different losses in the second step (Eq. (19)). The first row shows that the FL term alone degrades the performance. This is because training samples of a current dataset mainly contain new categories, causing a class imbalance between previous and new categories. The second row shows that replaying 1K features for each previous category lessens the influence of the class imbalance problem. From the last two rows, we can see that ALI is also helpful to alleviate the imbalance problem and it further boosts the performance along with replaying features.

Table 5: Comparison of IoU scores using different loss terms of our approach on 16-5(1) of PASCAL VOC [10].

(a)

CE	ALI	KD	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU
✓			17.43	6.41	14.81	9.35
✓	✓		74.66	46.83	68.04	57.55
✓	✓	✓	76.87	53.07	71.21	62.79

(b)

FL	ALI	MEM	$\text{mIoU}_{\text{base}}$	mIoU_{new}	mIoU	hIoU
✓			76.09	50.23	69.93	60.51
✓		✓	77.93	55.17	72.51	64.61
✓	✓		76.70	52.85	71.02	62.58
✓	✓	✓	78.04	55.34	72.63	64.76

5 Conclusion

We have introduced a new ISS method, ALIFE, that alleviates catastrophic forgetting and reduces memory requirements for an experience replay. First, we have presented a gradient analysis of CCE and CKD for better understanding the effects on catastrophic forgetting, and have proposed ALI incorporating the merits of CCE and CKD. Second, we have proposed a feature replay scheme using the Cayley transform that requires less memory footprint than memorizing raw images. Finally, we have shown that ALIFE sets a new state of the art on standard ISS benchmarks.

References

[1] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, 2018.

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

[3] Sungmin Cha, YoungJoon Yoo, Taesup Moon, et al. SSUL: Semantic segmentation with unknown label for exemplar-based class-incremental learning. *NeurIPS*, 2021.

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

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

[6] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[8] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. PODNet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, 2020.

[9] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. PLOP: Learning without forgetting for continual semantic segmentation. In *CVPR*, 2021.

[10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 88(2):303–338, 2010.

[11] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014.

[13] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *ICCV*, 2011.

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

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS Workshop*, 2014.

[16] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip HS Torr. Deeply supervised salient object detection with short connections. In *CVPR*, 2017.

[17] Qibin Hou, Li Zhang, Ming-Ming Cheng, and Jiashi Feng. Strip Pooling: Rethinking spatial pooling for scene parsing. In *CVPR*, 2020.

[18] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, 2019.

[19] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *ECCV*, 2020.

[20] Ronald Kemker and Christopher Kanan. FearNet: Brain-inspired model for incremental learning. *ICLR*, 2018.

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

[22] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *NeurIPS*, 2011.

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

[24] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[25] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *NeurIPS*, 2017.

[26] Andrea Maracani, Umberto Michieli, Marco Toldo, and Pietro Zanuttigh. RECALL: Replay-based continual learning in semantic segmentation. In *ICCV*, 2021.

[27] 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.

[28] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in psychology*, 4:504, 2013.

[29] Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. In *ICCV Workshop*, 2019.

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

[31] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *CVPR*, 2019.

[32] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. GDumb: A simple approach that questions our progress in continual learning. In *ECCV*, 2020.

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

[34] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *NeurIPS*, 2017.

[35] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *ACM SIGSAC CCS*, 2015.

[36] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, 2019.

[37] Ye Xiang, Ying Fu, Pan Ji, and Hua Huang. Incremental learning using conditional adversarial networks. In *ICCV*, 2019.

[38] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *CVPR*, 2020.

[39] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017.

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]
   (b) Did you describe the limitations of your work? [Yes] Please refer to the supplementary material.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (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? [Yes] We describe in Sec. 3.2 the assumption of CKD. Our ALI does not require any assumptions.
   (b) Did you include complete proofs of all theoretical results? [Yes] We provide detailed derivations in the supplementary material.
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)? [No] Our code and model will be made publicly available online, if the paper is accepted. However, please note that we present implementation details in Sec. 4.1 and more detailed descriptions in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We describe implementation details in Sec. 4.1. For more details, please refer to the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] For all experiments, we report averaged scores over 3 runs. For example, we report standard deviations in Tables 3 and 4.
   (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] We use 2 NVIDIA TITAN RTX GPUs for all experiments. Please refer to the supplementary material for details.
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? [Yes] We train DeepLab-V3 [5] on PASCAL VOC [10] and ADE20K [10].
   (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]