Replay Master: Automatic Sample Selection and Effective Memory Utilization for Continual Semantic Segmentation

Lanyun Zhu, Tianrun Chen, Jianxiong Yin, Simon See, De Wen Soh and Jun Liu, *Senior Member, IEEE*

***Abstract*—Continual Semantic Segmentation (CSS) extends static semantic segmentation by incrementally introducing new classes for training. To alleviate the catastrophic forgetting issue in this task, replay methods can be adopted, constructing a memory buffer that stores a small number of samples from previous classes for future replay. However, existing replay approaches in CSS often lack a thorough exploration of two critical issues: how to find the most suitable memory samples and how to utilize them for replay more effectively. Common strategies either randomly select samples or rely on hand-crafted, single-factor-driven methods that are hard to be optimal, and often employ conventional training techniques for replay that do not account for class imbalance problem resulting from limited memory capacity. In this work, we tackle these challenges by introducing a novel memory sample selection method that leverages a reinforcement learning framework with innovative state representations and a dual-stage action scheme to automatically learn a selection policy. Additionally, we propose an expert mechanism and a dual-phase training method to address the class imbalance issue, thereby enhancing the effectiveness of replay training by making better use of memory samples. Incorporating the proposed automatic sample selection and effective memory utilization methods, we develop a novel and effective replay-based pipeline for CSS. Our extensive experiments on Pascal VOC 2012 and ADE20K datasets demonstrate the effectiveness of our approach, which achieves state-of-the-art (SOTA) performance and outperforms previous advanced methods significantly.**

***Index Terms*—Continual semantic segmentation.**

I. Introduction

Semantic segmentation is an important task with numerous applications for both academic research and industrial applications. The rapid development of algorithms [14], [19], [64], [52] and the growing availability of large public datasets [22], [102], [25] have led to significant success in this field. Most existing methods for semantic segmentation are typically based on a setup where all classes are known beforehand and can be learned simultaneously. However, in many scenarios, this static setup is often unrealistic and does not meet practical needs, as the constantly changing environment may require the model to be continually updated to handle new classes. To address this challenge, continual semantic segmentation (CSS) [91], also known as class-incremental semantic segmentation, has been proposed and is gaining increasing attention in the research community. This task aims to adapt the previously learned model to accommodate newly added categories, thus allowing different classes to be learned incrementally at different stages.

A simplistic approach to implementing CSS is directly finetuning the old model with data from new classes. However, this method typically results in the model becoming biased towards the new classes, leading to a forgetfulness of previously learned information and a reduction in accuracy for old classes. This challenge, known as catastrophic forgetting, has become the biggest problem for CSS and prompted the development of various solutions. For example, some works [9], [55], [59], [57], [75], [92], [86] utilize knowledge distillation to incorporate the knowledge of the old model into the learning process of new classes. Other techniques [96], [81], [21] focus on multi-stage knowledge integration by merging the parameters of old and new models. Another category of methods [10], [85], [53], known as exemplar replay, employ a small subset of samples from previous classes to store into a memory, which are subsequently included in the training of future stages, thereby enhancing the model's ability to handle old classes and mitigating the issue of catastrophic forgetting. Demonstrated to be both robust and effective, such replay methods have been widely adopted in practical continual learning scenarios.

Due to the need to minimize storage usage and protect privacy, the memory capacity in these replay methods is typically limited, only allowing for the storage of a few selected samples. Therefore, the careful selection of the most suitable samples for replay becomes critical, which has received considerable attention by previous works [71], [63], [6], [1], [36], [5], leading to the development of various solutions aimed at finding the optimal method for sample selection. For instance, some researchers [54] advocate for selecting the most common samples with the lowest diversity, under the assumption that the most representative samples would boost the replay effectiveness better. However, these common samples may not necessarily be those most likely to be forgotten in subsequent stages, and thus might not always be the best choice for replay. Alternative approaches, such as the one suggested by [5], advocate for preserving both low-diversity samples near the distribution center and high-diversity samples near classification boundaries. This dual approach, however, introduces new challenges as the limited memory capacity makes it difficult to balance the

Lanyun Zhu and De Wen Soh are with Information Systems Technology and Design Pillar, Singapore University of Technology and Design, Singapore, 487372 (e-mail: lanyun_zhu@mymail.sutd.edu.sg, dewen_soh@sutd.edu.sg)

Tianrun Chen is with College of Computer Science and Technology, Zhejiang University, China, 310027 (email: tianrun.chen@zju.edu.cn)

Jianxiong Yin and Simon See are with NVIDIA AI Tech Centre, Singapore (e-mail: jianxiongy@nvidia.com, ssee@nvidia.com)

Jun Liu is with School of Computing and Communications, Lancaster University, UK (e-mail: j.liu81@lancaster.ac.uk)

Corresponding author: Jun Liu.

number of different types of samples effectively to maximize replay effectiveness. Moreover, most existing methods are designed based on a single factor, yet the performance of sample selection could be influenced by multiple factors with intricate interdependencies. For example, beyond diversity, the memory sample selection should also be class-dependent, as classes that are more challenging require a greater number of replay samples to effectively alleviate its more severe problem of catastrophic forgetting. Therefore, we argue that a more intelligent approach for memory sample selection is needed, one that could consider a broader spectrum of factors and their intricate interdependencies.

Witnessing the challenge, we argue that manually designing an optimal selection strategy is difficult due to the complicated factors that may affect replay effectiveness. To address this, we instead propose a novel mechanism that learns an optimal selection policy automatically and selects samples accordingly. Our key insight is that selecting memory samples can be viewed as a decision-making task across different training stages. Therefore, we formulate this operation as a Markov Decision Process and propose solving it automatically within a reinforcement learning (RL) framework. Specifically, we employ an agent network to make the selection decisions, which takes the state representation as input and selects optimal samples for replay. To enhance the decision-making effectiveness, we design a comprehensive, task-tailored state that integrates sample diversity and class performance features, in which a novel method for measuring similarity is introduced to compute this state more effectively and efficiently. We also propose a dual-stage action space where the agent not only selects the most appropriate samples for memory updates but also optimizes these samples for better replay effectiveness through gradient-based enhancements. These innovative designs constitute an effective RL mechanism, which was initially introduced in the conference version of this paper [107] and demonstrates enhanced sample selection capabilities that can significantly boost replay performance.

While the most suitable memory samples have been selected using our proposed method, how to utilize them better to achieve effective replay training remains a challenge, which we aim to address in this extended work. A primary concern is the limited number of samples retained for previous classes in the memory, which creates a significant disparity between the number of new and old class samples used for training. Such class imbalance could impair the model's ability to retain processing capabilities for older categories. To address this issue, in this extended work of our conference version [107], we further introduce a novel memory utilization method that incorporates a stage-specific expert mechanism with shared information usage. In this mechanism, the classes in each stage are assigned a dedicated expert, thus preventing all model parameters from being overly biased towards new classes with more training samples available. Additionally, we also enable selective cross-stage expert utilization to leverage the shared information across different categories. This enriches feature extraction pathways for each category and enhances the training effectiveness of the experts. Furthermore, we propose a dual-phase training paradigm consisting of a conventional training phase followed by a class-balanced finetuning phase, thereby addressing the class imbalance issue and further mitigating the catastrophic forgetting problem of old classes.

Incorporating the automatic sample selection and effective memory utilization methods proposed in this work, we develop a novel and effective replay-based pipeline for CSS, which simultaneously addresses two critical questions for the replay mechanisms in CSS: how to find the most suitable memory samples and how to utilize them for training more effectively. We evaluate our method on Pascal VOC 2012 and ADE20K, and the state-of-the-art (SOTA) performance on both of them demonstrates the superiority of our novel pipeline. Results indicate that by storing only 1% of the total training samples for replay, our method can achieve excellent CSS performance. To summarize, the contributions of this work are as follows:

- We develop an automatic selection mechanism to select the best replay samples enabled by a novel RL paradigm, which incorporates a task-tailored state representation containing comprehensive factors that can guide the selection decision, and a dual-stage action space to select samples and boost their replay effectiveness.
- We propose a novel method for the more effective utilization of memory samples, which incorporates a stage-specific expert mechanism with shared information utilization and a dual-phase training approach to overcome the class imbalance issue caused by the limited capacity of replay memory.
- Incorporating the proposed automatic sample selection and memory utilization methods, we develop a novel and effective replay-based pipeline for CSS, with extensive experiments on multiple datasets and protocols demonstrating its excellent performance.

As an extension of our previous conference paper [107] published on CVPR2023, this paper introduces significant enhancements in three key aspects. *First*, [107] focuses solely on how to select samples, but does not further explore how to leverage the selected samples more effectively for better replay training. This extended work addresses this crucial aspect by further proposing an innovative and effective memory utilization method (Section V), which includes a novel expert mechanism with shared information utilization, and a dual-phase training approach to alleviate the class imbalance issue. By incorporating these newly proposed methods, our extended pipeline presented in this paper achieves superior results than its conference version. *Second*, we expand our comparisons across a broader range of CSS settings and with additional methods (including transformer-based methods). The more extensive results provide stronger evidence of the advantages of our approach. (Section VI-B) *Third*, we conduct more comprehensive ablation study experiments and analysis to evaluate the effectiveness and soundness of our designs more thoroughly. (Section VI-C)

II. Related Work

1) Image Segmentation: Image segmentation is a crucial research area in computer vision. In recent years, the rapid development of deep learning has led to significant advancements in this field. Techniques based on fully convolutional

networks [52], UNet architecture [64], [3], pyramid networks [14], [100], dilated convolution [12], [13], [14], [15], attention mechanisms [26], [110], [108], and transformers [83], [19], [20], [32], [60] have established a new paradigm for tasks such as semantic segmentation [14], [83], [109], [20], [100], [45], [34], [79], instance segmentation [30], [42], [11], [37], [77], [82], panoptic segmentation [70], [38], [18], [16], [56], [46], and video object segmentation [7], [97], [88]. Recently, some methods [104], [44], [31] explore improved representation spaces for semantic segmentation. For example, [104] proposes a non-parametric framework that replaces traditional learned classifiers with multiple class prototypes derived via online clustering, enabling efficient and interpretable segmentation through nearest-prototype inference. [44] introduces a novel segmentation framework that integrates symbolic reasoning with neural networks to better mimic human-like structured visual understanding. Other methods [103], [47], [78], [89], [106] focus on more effective training strategies for segmentation. For instance, [103] pioneers the application of pixel-wise contrastive learning to supervised semantic segmentation, achieving significant performance improvements. [47] proposes a novel generative-discriminative hybrid training scheme by modeling class-conditional feature distributions using Gaussian Mixture Models. More recently, approaches based on foundation models, such as CLIP [61], [48], [35], Segment Anything Models (SAM) [39], [93], [17], and multimodal large language models [41], [62], [106], [105], have further advanced segmentation performance by leveraging the rich pre-trained knowledge embedded in foundation models, thereby achieving strong generalization capabilities. Despite their success, most of these methods operate under a conventional setup where all training data are known beforehand and can be learned simultaneously. In contrast, our work focuses on the more practical and challenging continual learning setting for semantic segmentation.

2) Continual Segmentation: Continual segmentation overcomes the limitations of traditional segmentation algorithms by allowing different classes to be learned at different stages. To address the biggest challenge in this task—catastrophic forgetting—some methods [9], [55], [59], [57], [75], [92] adopt knowledge distillation techniques that enable the model to review knowledge from previous stages during the learning process of new stages. For example, MIB [9] addresses the challenge of semantic shift in the background class by proposing a novel weight initialization method and distillation loss. SDR [55] proposes a novel distillation-based preservation technique based on prototype matching, contrastive learning, and feature sparsity. CSW-KD [59] selectively revises the knowledge of old classes that are likely to be forgotten through distillation by focusing on those that are visually similar to the new classes. LGKD [87] proposes a label-guided knowledge distillation loss to mitigate the issue of confusion between the background and novel classes in the continual scenarios. Other methods [24], [98], [50], [72] address the forgetting problem by annotating the new stage's data with pseudo-labels for the old stage's classes. For example, PLOP [24] employs a pseudo-labeling mechanism based on prediction confidence to improve continual learning performance. CoinSeg [98] introduces a contrastive-learning-based distillation to enhance both inter-class and intra-class representations with the guidance of pseudo-labels. Some methods [96], [81], [21] use weight fusion to combine model parameters from both new and old stages to retain previous knowledge. For example, EWF [81] fuses models from different stages by taking the weighted average of their corresponding parameters. Cs2K [21] further improves the model fusion method by selectively integrating different model parameters through weight-guided selective consolidation. Another effective approach to addressing catastrophic forgetting is to employ a portion of past data for replay. For example, some methods [10], [85] use a memory buffer to store replay exemplars, but the samples are selected either randomly or based on heuristic rules. [53] derives richer replay exemplars through a generative adversarial network, but this approach comes with high computational costs and requires additional web-crawled images. Our work also adopts a memory replay mechanism but with a different and entirely new pipeline, in which we introduce a novel automatic sample selection mechanism and propose a new, effective replay training method to utilize memory samples better. Some more recent works [65], [58], [8], [28] are designed based on transformer networks. We have also conducted experiments to compare with these methods and achieved superior results. (Please see Section VI-B2 for details)

3) Memory Sample Selection: As an important approach to addressing catastrophic forgetting in continual learning, the effectiveness of replay-based methods relies on the careful selection of memory samples. Most of the previous selection methods are designed based on hand-crafted heuristic rules. For example, some methods [1], [90], [36], [5] consider sample diversity as a key factor in determining replay effectiveness and select samples accordingly. [66] uses adversarial Shapley value for sample selection to preserve latent decision boundaries for previously observed classes. [63] proposes selecting a fixed number of representative samples that best capture the feature distribution of each class. Despite some success, such hand-crafted methods are difficult to be optimal due to the complex interplay between different factors that can affect selection performance. In contrast, our method explores a novel direction by enabling the selection policy to be automatically learned through a carefully designed RL mechanism, which achieves better performance than previous selection methods as demonstrated by the experimental results presented in Section VI-B4.

4) Reinforcement Learning: Reinforcement learning (RL) is a technique that has achieved remarkable success in many decision-making tasks, such as game intelligence [67], robot control [68], and large language models [4]. It has also been applied to computer vision in various areas such as active learning [27], pose estimation [29], model compression [2], and person re-identification [80]. A previous work related to our method is [51], which uses RL for exemplar length management but operates with a completely different mechanism from ours. Instead of employing RL to control class-level memory length while still relying on a random selection process as in [51], our method is end-to-end and can directly select specific samples in a fully automated, single step. In

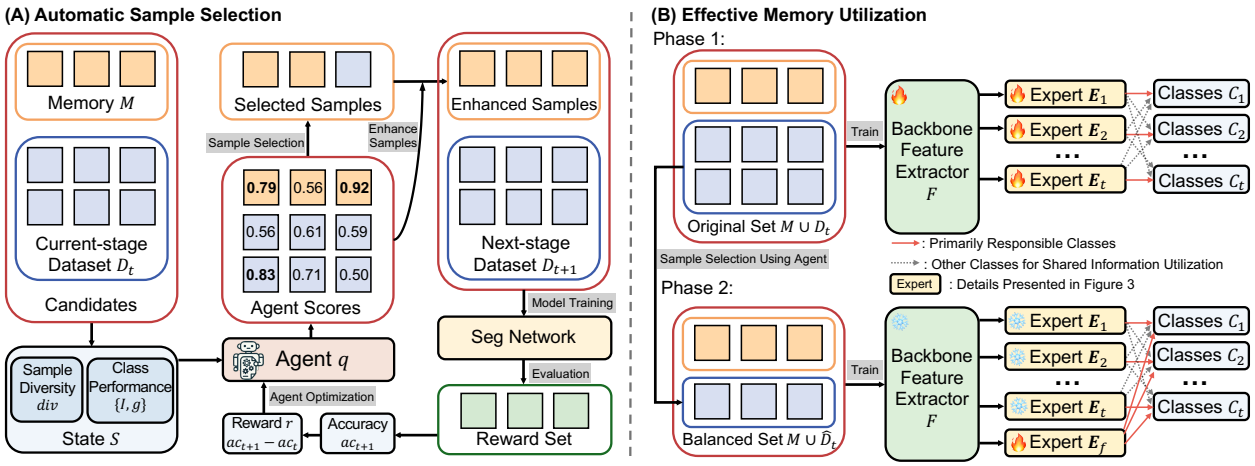


Fig. 1. The overall framework of RPMaster, including (A) Automatic Sample Selection (Section IV), and (B) Effective Memory Utilization (Section V).

addition to the selection method, we also propose a novel approach to better utilize the samples selected by the RL policy. By incorporating these designs, our method introduces a brand-new RL-based replay pipeline for CSS.

III. Preliminaries

Continual semantic segmentation (CSS) aims to train a segmentation model in T stages continuously without forgetting. In each stage t, a training dataset $\mathcal{D}_t$ can be utilized, where only pixels within the current classes $\mathcal{C}_t$ are labeled, while pixels from other classes (including previous classes $\mathcal{C}_{1:t-1}$ and future classes $\mathcal{C}_{t+1:T}$) are treated as the background. The objective is to allow the model to be able to predict all classes $\mathcal{C}_{1:T}$ after completing all T stages. To alleviate the catastrophic forgetting problem in CSS, we follow the setting of previous replay-based methods [10], [85], where an exemplar memory $\mathcal{M}$ that contains a small number of sampled data from the previous classes can be used for replay. This allows both $\mathcal{M}$ and $\mathcal{D}_t$ to be involved in training at stage t.

During the training process, $\mathcal{M}$ is updated once a training stage is completed. This means $\mathcal{M}$ will be refilled by new samples from $\mathcal{M} \cup \mathcal{D}_t$ after the stage t is completed. It is obvious that the careful selection of samples for $\mathcal{M}$ and their effective utilization for training can significantly impact performance, which is also the primary focus of this work. In the following Section IV and Section V, we will illustrate our solutions for addressing the challenges in memory sample selection and memory sample utilization, respectively.

IV. Automatic Sample Selection

A. Overall

Considering the memory $\mathcal{M}$ with L samples and $\mathcal{D}_t$ with N_t samples, the target of this work is to learn an optimal policy that automatically selects L samples from $\mathcal{M} \cup \mathcal{D}_t$ and put them into $\mathcal{M}$ for training in the next stage, driven by the goal of maximizing a reward that reflects performance improvement. The selection decision is made by an agent network q, which is a three-layered MLP. This network transforms the learning of the selection policy into a decision-making process with the following procedure: 1) Obtaining the state s by assessing the properties of samples that can measure its contribution for replay. 2) Based on s, using the agent q to make an action a that selects L samples to update the memory $\mathcal{M}$. 3) Training the segmentation network with the updated $\mathcal{M}$. 4) Computing the reward r based on the validation performance of the updated segmentation network. 5) Repeating the above steps until completing all T stages. 6) Optimizing agent q based on r from all stages.

As shown in Figure 1(A), in this work, we solve the above problem using a reinforcement learning (RL) framework, in which the agent q evaluates each state s and decides on an action a based on this evaluation. By leveraging the proposed task-tailored state representations, a novel dual-stage action for selection enhancement and the reward-driven optimization, we enable the agent to learn an effective selection policy that can enhance the replay performance. In the following sections, we illustrate the details of how these components are designed.

B. State Representation

The state representation s is the key to enabling the automatic selection decision process effective, as it serves as the input to and the decision support for the agent network. Designing an effective state representation should consider and align with the requirements of the selection policy. Intuitively, an optimal policy should make selection decisions by estimating the potential replay contribution of each sample and allocating different quotas to different classes, as harder classes suffer more from catastrophic forgetting and thus require more samples for replay. Based on these intuitions, we propose combining two key factors—***sample diversity*** and ***class performance***—to construct the state representation. For an image within class c, sample diversity div is a factor that measures its novelty, which can reflect the potential replay effectiveness as suggested by previous works [5], [63]. We calculate this factor by computing and averaging inter-sample similarities, with a higher div indicating that the sample differs more from other images within the same class c.

Class performance is constructed using a combination of two metrics: 1) accuracy and 2) forgetfulness. Accuracy is derived by computing the training IoU I_c for each class c, with harder classes that achieve poorer performance having lower IoUs. However, while this IoU factor measures the current training accuracy for each class, it cannot reflect whether a class is likely to be forgotten in the future, which is a critical consideration for CSS but difficult to measure directly because future performance is unknown. To address this problem, we estimate forgetfulness g_c by measuring the similarities between class c with all other classes, motivated by the previous finding that classes more similar to others are more likely to be forgotten [59]. Eventually, for a given image, we compute the diversity $\{div_c\}_{c=1}^{C}$, accuracy $\{I_c\}_{c=1}^{C}$, and forgetfulness $\{g_c\}_{c=1}^{C}$ for all C classes present in the image, resulting in three groups of features. We then calculate the average values of these three groups across different classes and concatenate them to form the state representation s of the image.

C. Measuring Similarity in Multi-structure Space

1) Motivation: Both the sample diversity div and forgetfulness g_c introduced above require the computation of similarity between different images. In previous works, this similarity is primarily measured in either the *prototype-level space* [63] or *pixel-level space* [76]. The prototype-level approach condenses the feature map of a sample into a single prototype and then calculates inter-prototype distances. While computationally efficient, this method sacrifices spatial information and structural details, which may lead to errors. For instance, two images with entirely different local structures or object postures may have similar prototypes, as these prototypes are computed by averaging all pixel features, thereby concealing differences in local details. Such errors caused by the lack of local detail are detrimental to segmentation tasks, since the local structural information is crucial for accurate pixel-level predictions as indicated by previous works [109], [33]. The pixel-level approach retains local information but incurs an unacceptable computational cost due to the pixel-wise distance calculations and may lead to overfitting [43]. Therefore, to obtain a more informative similarity metric, a novel representation space is needed—one that can preserve spatial and structural information while remaining computationally feasible. To address this, we propose a novel method that first maps each sample into a *multi-structure graph space* and then measures inter-sample similarity based on graph matching. In this graph, each vertex represents a semantic structure, and each edge represents the spatial and semantic correlations. In this way, a fine-grained similarity can be calculated by utilizing the comprehensive information in the graph, while requiring only a small amount of computation benefiting from the condensed feature representation.

2) Multi-structure Graph: Considering an image with class c, we represent the region $\mathcal{R}$ within c as a graph $\mathcal{G}$ through the way illustrated by Figure 2. To obtain the local structural representation, we first employ the method in [73] to produce M superpixels $\{r_m\}_{m=1}^{M}$ $(r_1 \cup r_2 \cup ... \cup r_M = \mathcal{R})$. The motivation for generating superpixels is that, each produced

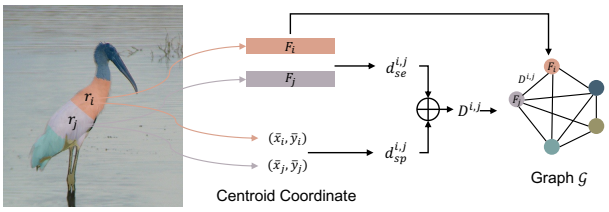


Fig. 2. Illustration of how the graph for computing sample diversity is constructed. In the figure, r_i and r_j denote two superpixels. F_i and F_j refer to the average features for all pixels within them. $(\overline{x}_i, \overline{y}_i)$ and $(\overline{x}_j, \overline{y}_j)$ denote the centroid coordinates of r_i and r_j respectively. $d_{se}^{i,j}$ and $d_{sp}^{i,j}$ refer to the semantic distance and spatial distance. The generated graph $\mathcal{G}$ will be used to compute the sample diversity and forgetfulness.

r_m can represent a meaningful semantic structure, such as the head of a bird; and it can also condense the pixel-level representation by clustering pixels with similar features and adjacent positions, thus enabling the subsequent processes to be completed more efficiently with reduced computational cost. Each vertex F_m is then computed as the average backbone feature for all pixels within r_m. We represent the edges of $\mathcal{G}$ as a distance map $D \in \mathbb{R}^{M \times M}$, where each element $D^{i,j}$ denotes the distance between the i-th and j-th vertices. To simultaneously consider the context-aware high-level semantic information and low-level spatial correlation, we combine both the *semantic distance* and *spatial distance* to get D. Specifically, the semantic distance $d_{se}^{i,j}$ is the L2 distance between F_i and F_j; while the spatial distance $d_{sp}^{i,j}$ is the Euclidean distance between the two centroid coordinates [1] of the superpixels r_i and r_j, which reflect the relative positions of them. We normalize $d_{se}^{i,j}$ and $d_{sp}^{i,j}$ to $[0, 1]$ and derive $D^{i,j} = d_{se}^{i,j} + d_{sp}^{i,j}$. Such a graph captures comprehensive representations of an image, such as the local structural details and spatial information, which are lost in the prototype space but are crucial for measuring a fine-grained similarity.

3) Inter-graph Similarity: After mapping each image into the aforementioned graph space, we then measure the inter-image similarities using a matching algorithm. Specifically, for two graphs $\mathcal{G}_i$ and $\mathcal{G}_j$ from two images, we apply the Sinkhorn algorithm [23] to align them, through which the transport cost tc is obtained by solving the optimal transport problem, with a higher tc indicating the lower similarity between these two graphs. The details for this step are presented in Appendix A. Since the edge distance $D^{i,j}$ is computed using both semantic and spatial distances, the resulting tc after matching can reflect both semantic and spatial similarity. For example, when comparing two regions of the 'person' class, we can assess both whether they are wearing similar clothes (semantic similarity) and whether they have the same body posture (spatial similarity), thereby capturing comprehensive fine-grained representations that contribute to an effective state representation in our framework.

4) Representation Computation: We use the aforementioned similarity measurement to compute sample diversity div and forgetfulness g in our state representations. For an

[1] Considering a superpixel $r = \{(x_i, y_i)\}_{i=1}^{N}$, the centroid coordinate $(\overline{x}, \overline{y})$ is computed as: $\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$, $\overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$.

image containing the c-th class, let $\mathcal{G}$ represent its graph. We introduce a support set $\mathcal{S}_c = \{\mathcal{G}_c^i\}_{i=1}^{N_c}$ for computing div, which contains several graphs of other images within the same class c. For each previous class $c \in \mathcal{C}_{1:t-1}$, $\mathcal{S}_c$ is constructed as the set of all images for c stored in the memory $\mathcal{M}$. While for each current class in $\mathcal{C}_t$ that has a larger number of images, we randomly sample 10% of all images to form $\mathcal{S}_c$ for reducing the computational burden. Our experimental results presented in Section VI-C5 demonstrate that div computed from such a sampled set is effective enough. A diverse and novel sample is likely to have low similarities compared to other samples within the same class. Therefore, we calculate div by computing the average similarities as follows:

$$div = \frac{1}{|\mathcal{S}_c|} \sum_{\mathcal{G}_c^i \in \mathcal{S}_c} \mathrm{Sim}\left(\mathcal{G}, \mathcal{G}_c^i\right), \tag{1}$$

where Sim refers to the inter-graph similarity measurement introduced in Section IV-C3, which outputs the transport cost of mapping $\mathcal{G}$ and $\mathcal{G}_c^i$ (please see Appendix A for details). A higher value of $\mathrm{Sim}\left(\mathcal{G}, \mathcal{G}_c^i\right)$ indicates a lower similarity between graphs $\mathcal{G}$ and $\mathcal{G}_c^i$. To calculate forgetfulness g_c for each class c, we first construct a representative set $\hat{\mathcal{S}}_c = \{\mathcal{G}_c^i\}_{i=1}^{\hat{N}_c}$, which contains the top 10% of samples in $\mathcal{S}_c$ with the lowest diversity scores div. These samples are most similar to other samples in c and can therefore serve as representatives for the entire class. The forgetfulness g_c is then computed as the class-wise similarity as follows:

$$g_c = \frac{1}{|\hat{\mathcal{S}}_c|} \sum_{\mathcal{G}_c^i \in \hat{\mathcal{S}}_c} \frac{1}{|\mathcal{C}_{1:t}| - 1} \sum_{j \in \mathcal{C}_{1:t} \setminus c} \frac{1}{|\hat{\mathcal{S}}_j|} \sum_{\mathcal{G}_j^k \in \hat{\mathcal{S}}_j} \mathrm{Sim}\left(\mathcal{G}_c^i, \mathcal{G}_j^k\right). \tag{2}$$

Eventually, the obtained div and g are combined with the accuracy I, generating the state representations that can help make a wiser selection decision.

D. Dual-stage Action with Sample Selection and Enhancement

After obtaining the state information s^i for each sample, we use an agent network q to produce a score $q(s^i)$ by taking s^i as the input. A higher score indicates that this sample is more suitable for replay. Thus, we consider the agent's output score as an indicator of replay effectiveness and use it to drive a novel action space for the RL mechanism, which operates in two stages: *sample selection* and *sample enhancement*.

Specifically, we first select memory samples by choosing the top L samples with the highest agent scores, expressed as:

$$a = \underset{i \in [1, L+N_t]}{\mathrm{TopL}} \; q\left(s^i\right). \tag{3}$$

After that, instead of directly using the static selected samples for training in the next stage, we further propose an enhancement operation that edits each sample to be more effective for replay. This approach is motivated by our observation of the agent scores for the selected samples. Specifically, we noticed that only 10% of the selected samples have agent scores exceeding 0.8 (with the theoretical maximum score being 1). This indicates that while these samples are the best possible choices among the available candidates, they are still not the ideally perfect samples for replay. Thus, we implement enhancement through a gradient-based approach by maximizing the agent score. Specifically, we treat the state s^x as a feature computed from the input image x, along with $\mathcal{M}$ and $\mathcal{D}_t$, under the segmentation network parameters θ_{seg} with the state computing function f_s, which is formulated as:

$$s^x = f_s\left(x; \mathcal{M}, \mathcal{D}_t, \theta_{seg}\right). \tag{4}$$

The agent score is then generated as $q(s^x)$. We perform a gradient update on x to increase the agent score $q(s^x)$ and thereby enhancing the replay effectiveness. This process is formulated as:

$$\begin{aligned} x' &= x + \epsilon \nabla_x q\left(s^x\right) \\ &= x + \epsilon \nabla_x q\left(f_s\left(x; \mathcal{M}, \mathcal{D}_t, \theta_{seg}\right)\right), \end{aligned} \tag{5}$$

where ϵ is a hyper-parameter to control an adequate updating rate to ensure that the image label remains unchanged. With the higher agent score, the resulting x' can be more effective and is stored into $\mathcal{M}$ for replay.

E. Reward and Optimization

Our selection policy aims to enable the segmentation model trained with the memory $\mathcal{M}$ to achieve better performance. Therefore, the reward for optimizing the agent should reflect how much the memory samples derived by the agent's policy can benefit the CSS training. To implement this, we divide a subset from the training set to get a reward set $\mathcal{D}^{reward}$, compute the validation accuracy ac_t for the segmentation model that has completed the t-th stage on $\mathcal{D}^{reward}$, and obtain reward r_t by $r_t = ac_t - ac_{t-1}$. With the reward derived, following DQN [74], the agent is optimized by the temporal difference (TD) error formulated as:

$$\begin{aligned} TD\left(\theta, \hat{\theta}\right) = \frac{1}{T-1} \sum_{t=1}^{T-1} \Bigg(r_{t+1} + \frac{\gamma}{L} \sum_{i=1}^{L} q\left(s_{t+1}^{a_{t+1}^i}; \hat{\theta}\right) \\ - \frac{1}{L} \sum_{i=1}^{L} q\left(s_t^{a_t^i}; \theta\right) \Bigg)^2, \end{aligned} \tag{6}$$

where $s_t^{a_t^i}$ refers to the state representation of the i-th selected sample in the t-th stage, θ and $\hat{\theta}$ refer to the agent's policy and off-policy parameters respectively. Following [74], $\hat{\theta}$ is periodically updated based on θ, aiming to save the learned Q-value.

F. Agent Training and Deployment

With the RL mechanism for CSS introduced above, we then present the agent training and deployment method in this section. We denote $\mathcal{D}_1$ as the dataset for the first-stage training. According to the CSS protocol [24], $\mathcal{D}_1$ contains multiple classes (usually more than half of the total), thus it can provide sufficient information for training an effective agent. The detailed training process of the agent is shown in Alg. 1. Specifically, we train the agent for Y iterations. In each iteration, we randomly divide $\mathcal{D}_1$ into a training set $\mathcal{D}_1^{train}$ and a reward set $\mathcal{D}_1^{reward}$, and set a new CSS task by reallocating

Algorithm 1 Agent Training Algorithm.

Input: agent network q, segmentation network parameters θ_{seg}, dataset $\mathcal{D}_1$.
for y **in** $1, ..., Y$ **do**
  Randomly create a new task having T_y continual stages with class partitions $\{\mathcal{C}_{t_y}\}_{t_y=1}^{T_y}$.
  Partition $\mathcal{D}_1$ to $\mathcal{D}_1^{train}$ and $\mathcal{D}_1^{reward}$
  Initialize θ_{seg}, initialize $\mathcal{M}$ as an empty set
  for t_y **in** $1, ..., T_y$ **do**
    Train θ_{seg} on $\mathcal{M} \cup \mathcal{D}_1^{train,t_y}$
    Compute state s^x (Sec.IV-B) and agent score $q(s^x)$ for every sample $x \in \mathcal{M} \cup \mathcal{D}_1^{train,t_y}$.
    Select and enhance samples (Sec.IV-D), update $\mathcal{M}$
    if $t_y > 1$ **then**
      Compute reward r_{t_y} using $\mathcal{D}_1^{reward}$ (Sec.IV-E)
    end if
  end for
  Update q by Eq. 6
end for
Return: q

the classes observed in each stage. This helps the agent to learn a more general policy by training from diverse settings.

Once the agent training is completed, we can deploy it on the whole set $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^{T}$, selecting and enhancing memory samples at the end of each stage and using them for replay in the next stage.

V. Effective Memory Utilization

A. Motivation

After training an agent using the aforementioned method and employing it to select samples at the end of each stage $t-1$, the next challenge is how to effectively utilize these samples for replay training in the next stage t. Due to the limited capacity of memory $\mathcal{M}$, the number of samples used for training $\mathcal{C}_{1:t-1}$ in stage t is significantly less than those for training $\mathcal{C}_t$. This class imbalance issue could cause the model to be overly biased towards classes $\mathcal{C}_t$ with much more training samples, thereby hindering its effectiveness in learning segmentation abilities for classes $\mathcal{C}_{1:t-1}$. Some previous methods assign pseudo-labels of classes $\mathcal{C}_{1:t-1}$ to the dataset $\mathcal{D}_t$ and thereby increasing $\mathcal{C}_{1:t-1}$'s training samples. However, to ensure accuracy, these methods typically select only high-confidence samples for pseudo-labeling. As a result, the increase in the number of training samples for $\mathcal{C}_{1:t-1}$ is very limited, and the class imbalance issue still remains.

To address the aforementioned issue, a straightforward approach is to assign a separate expert $\{\boldsymbol{E}_i\}$ to each $\mathcal{C}_i \in \mathcal{C}_{1:t}$, with each expert responsible only for extracting features relevant to the segmentation of its corresponding classes $\mathcal{C}_i$, rather than sharing a common set of parameters across all classes. This design aims to prevent all model parameters from being biased toward $\mathcal{C}_i$, which typically has more training samples than previous classes. However, although this naive method yields some improvements, we found its performance to still not be sufficiently satisfactory. This may be due to the limited number of training samples available for $\{\mathcal{C}_i\}_{i=1}^{t-1}$ in $\mathcal{M}$, which makes it difficult to effectively train their corresponding experts $\{\boldsymbol{E}_i\}_{i=1}^{t-1}$ using such insufficient data.

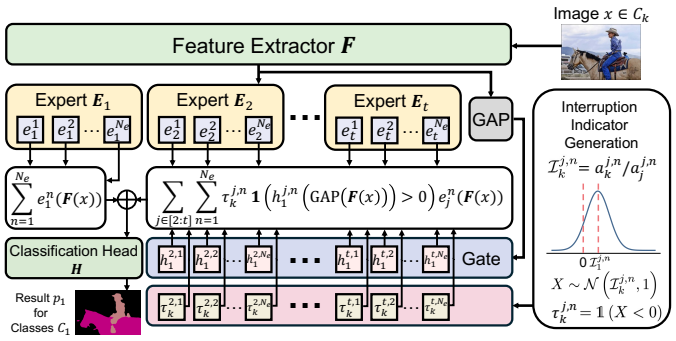


Fig. 3. Illustration of our expert mechanism with shared information utilization in the t-th training stag. GAP denotes global average pooling. For simplicity, we only present the process of segmenting classes $\mathcal{C}_1$ for an input sample within $\mathcal{C}_k$. The segmentation processes for other classes are similar.

The above discussion suggests that neither extreme cross-class sharing nor complete expert isolation is optimal. Instead, we propose a novel method that strikes a balance between the two: each expert $\boldsymbol{E}_i$ is primarily responsible for a specific $\mathcal{C}_i$, but is also allowed to selectively learn useful shared information from other classes $\{\mathcal{C}_j\}_{j\in[1:t]\setminus i}$ under a controlling strategy. This design helps mitigate overfitting while also enabling more effective training of the expert modules. In the following section, we present the details of this novel method.

B. Expert with Shared Information Utilization

Specifically, as shown in Figure 3, instead of using a single network layer as the expert $\boldsymbol{E}$, we draw inspiration from MOE [94] and set up N_e parallel sub-experts $\{\boldsymbol{e}_i^n\}_{n=1}^{N_e}$ to form $\boldsymbol{E}_i$. Each $\boldsymbol{e}_i^n$ is implemented as a 3×3 convolutional layer with 128 output channels. Different sub-experts receive the same backbone features as input, but are encouraged to extract different information during training (see Section V-C for details). During the segmentation of classes $\mathcal{C}_i$ for an input sample x, besides leveraging its corresponding expert $\boldsymbol{E}_i = \{\boldsymbol{e}_i^n\}_{i=1}^{N_e}$, the other sub-experts are also selectively utilized for feature extraction under the control of an activation gate. This gate is determined by a fully connected layer $h_i^{j,n}$, which receives the global average pooling of the backbone feature $\boldsymbol{F}(x)$ and produces a score that indicates whether the sub-expert $\boldsymbol{e}_j^n$ should be activated for using. The prediction result p_i for classes $\mathcal{C}_i$ can then be formulated as:

$$p_i = \boldsymbol{H}\left(\sum_{n=1}^{N_e} \boldsymbol{e}_i^n\left(\boldsymbol{F}\left(x\right)\right) + \sum_{j\in[1:t]\setminus i}\sum_{n=1}^{N_e} \mathbb{1}\left(h_i^{j,n}\left(\mathrm{GAP}\left(\boldsymbol{F}\left(x\right)\right)\right) > 0\right) \boldsymbol{e}_j^n(\boldsymbol{F}(x))\right), \tag{7}$$

where GAP refers to global average pooling. Considering the significant difference in the number of training samples for different $\mathcal{C}_i$, we aim to prevent each expert $\boldsymbol{E}_i$ from learning segmentation capabilities from an excessive number of samples of other classes, which could cause an imbalance issue. To address this, during the training phase, we introduce

an imbalance factor for generating a probability to randomly interrupt the use of each sub-expert $\boldsymbol{e}_i^n \in \{\boldsymbol{e}_i^n\}_{n=1}^{N_e}$ of $\boldsymbol{E}_i$ when processing input samples that do not belong to their respective classes $\mathcal{C}_i$. Specifically, in the m-th iteration of the training process at stage t, we count the number of times each $\boldsymbol{e}_j^n$ has been activated when processing samples within each $\mathcal{C}_k$ [2] across all previous $m-1$ iterations, denoted as $a_k^{j,n}$. The imbalance factor $\mathcal{I}_k^{j,n}$ of $\boldsymbol{e}_j^n$ for $\mathcal{C}_k$ is then calculated as $\mathcal{I}_k^{j,n} = a_k^{j,n}/a_j^{j,n}$. During training, given a sample within $\mathcal{C}_k$, the prediction process of p_i in Eq.7 is rewritten as:

$$p_i = \boldsymbol{H}\left(\sum_{n=1}^{N_e} \boldsymbol{e}_i^n\left(\boldsymbol{F}\left(x\right)\right) + \sum_{j\in[1:t]\setminus i}\sum_{n=1}^{N_e} \tau_k^{j,n} \mathbb{1}\left(h_i^{j,n}\left(\mathrm{GAP}\left(\boldsymbol{F}\left(x\right)\right)\right) > 0\right) \boldsymbol{e}_j^n(\boldsymbol{F}(x))\right), \tag{8}$$

where

$$\tau_k^{j,n} = \mathbb{1}\left(X < 0\right); X \sim \mathcal{N}\left(\mathcal{I}_k^{j,n}, 1\right), \tag{9}$$

where $\mathcal{N}$ denotes the Gaussian distribution. Through this method, classes that have excessively activated sub-expert $\boldsymbol{e}_j^n$ in previous training iterations — i.e., with more activation counts compared to samples from $\boldsymbol{e}_j^n$'s corresponding classes $\mathcal{C}_j$ — are less likely to use $\boldsymbol{e}_j^n$ for their training samples in the current iteration. This encourages each $\boldsymbol{e}_j^n$ to learn segmentation capabilities from a more balanced set of samples, thereby balancing the number of negative samples encountered by the expert of each class and effectively enhancing performance.

C. Dual-Phase Model Training

Building on the novel network structure introduced above, we further elaborate on the training details of our method in each stage t. To further address the issue of class imbalance, we propose a dual-phase training mechanism that consists of a conventional training phase for learning segmentation abilities followed by a finetuning phase to achieve class balance. In the first phase, all model parameters are trained using all samples from $\mathcal{M} \cup \mathcal{D}_t$. The loss function for this phase is written as:

$$\mathcal{L} = \mathcal{L}_{seg} + \mathcal{L}_{div} \tag{10}$$

where $\mathcal{L}_{seg}$ is the cross-entropy loss for segmentation training, in which we also employ the pseudo label mechanism as in [24] to improve accuracy. $\mathcal{L}_{div}$ enforces different sub-experts $\{\boldsymbol{e}_i^n\}_{n=1}^{N_e}$ in each $\boldsymbol{E}_i$ to extract diverse and orthogonal features, which is formally written as:

$$\mathcal{L}_{div} = \frac{1}{tN_e^2}\sum_{i=1}^{t}\sum_{n=1}^{N_e}\sum_{m=1}^{N_e}\frac{\boldsymbol{e}_i^n\left(\boldsymbol{F}\left(x\right)\right)\left(\boldsymbol{e}_i^m\left(\boldsymbol{F}\left(x\right)\right)\right)^{\mathrm{T}}}{\|\boldsymbol{e}_i^n\left(\boldsymbol{F}\left(x\right)\right)\|\|\boldsymbol{e}_i^m\left(\boldsymbol{F}\left(x\right)\right)\|}. \tag{11}$$

In the second phase, to enhance class balance, instead of using all samples in $\mathcal{M}\cup\mathcal{D}_t$ for training, we use the proposed memory selection method to select S images from $\mathcal{D}_t$ and combine them with $\mathcal{M}$ to form a more balanced training set $\mathcal{M} \cup \hat{\mathcal{D}}_t$, where S equals the average number of samples per $\mathcal{C}_i$ in $\mathcal{M}$. Given the small number of samples in $\mathcal{M} \cup \hat{\mathcal{D}}_t$, to avoid overfitting, we do not use this dataset to perform full finetuning on the whole model. Instead, we introduce an additional expert $\boldsymbol{E}_f$ with N_e sub-experts in parallel with the existing experts $\{\boldsymbol{E}_i\}_{i=1}^t$. The prediction process for all classes $\mathcal{C}_{1:t}$ can use each sub-expert in $\boldsymbol{E}_f$ for feature extraction. During fine-tuning, only $\boldsymbol{E}_f$ is updated, while all other model parameters are kept frozen. The loss function for this phase is the same as in the first phase (Eq.10). After fine-tuning, $\boldsymbol{E}_f$ can be merged with each $\boldsymbol{E}_i \in \{\boldsymbol{E}_i\}_{i=1}^t$ to form the new $\boldsymbol{E}_i$ used for the next continual stage. By adopting this dual-phase training strategy, our model not only learns segmentation capabilities effectively from abundant data, but also distributes these capabilities more evenly across different classes through fine-tuning with a class-balanced set of positive and negative samples. Experimental results in Section VI-C6 demonstrate that our method outperforms using either single phase alone.

[2]Here, the $\mathcal{C}_k$ to which each sample belongs is determined by the categories indicated in its label. In practice, since pseudo-labels are employed during training, a sample may belong to multiple $\mathcal{C}_k$. In such cases, we choose the largest $\mathcal{I}_k^{j,n}$ from all the $\mathcal{C}_k$ the sample belongs to for the calculation in Eq.9.

VI. Experiments

A. Experimental Settings

1) Datasets and CSS Protocols: We evaluate our method on two popular semantic segmentation datasets – Pascal VOC 2012 [25] and ADE20K. Pascal VOC 2012 is a large object segmentation dataset containing 10582 images for training and 1449 images for testing, with 20 foreground classes in total. ADE20K [102] is a large dataset for scene parsing. It has 150 foreground classes, with 20210 and 2000 images for training and testing, respectively. There are two different settings used in the research field of CSS: *Disjoint* and *Overlapped*. The key difference between them lies in whether the training set $\mathcal{D}_t$ at each stage t contains pixels belonging to other classes in $\mathcal{C}_{1:t-1}$ and $\mathcal{C}_{t+1:T}$. Specifically, in the *Disjoint* setting, the samples in $\mathcal{D}_t$ contain only pixels belonging to classes $\mathcal{C}_t$. In contrast, in the *Overlapped* setting, the samples in $\mathcal{D}_t$ can contain pixels belonging to any classes from $\mathcal{C}_{1:t-1} \cup \mathcal{C}_t \cup \mathcal{C}_{t+1:T}$. Note that in both settings, pixels in $\mathcal{D}_t$ that do not belong to $\mathcal{C}_t$ are labeled as the background category. Experimental results on both settings are reported and compared. To evaluate the effectiveness of our method comprehensively, we conduct experiments across multiple CSS protocols. Each protocol is denoted as '$m-n(T$ stages)', which refers to the model being trained over a total of T continual stages, in which the first stage $\mathcal{C}_1$ includes m classes while each of the remaining $T-1$ stages includes n classes. In this work, following the settings of previous works, we conduct experiments based on 19-1(2 stages), 15-5(2 stages) and 15-1(6 stages) for Pascal VOC 2012, and 100-50(2 stages), 100-10(6 stages), 100-5(11 stages) and 50-50(3 stages) for ADE20K.

2) Metrics: We use mIoU as the metric to evaluate the model's performance, which is widely used by most previous CSS methods and other semantic segmentation works. Following [24], [92], we compute the mIoU for the classes in the first stage $\mathcal{C}_1$, for the classes in the remaining continues stages $\mathcal{C}_{2:T}$, and for all classes $\mathcal{C}_{1:T}$ in all stages. These metrics respectively reflect the model's ability to overcome catastrophic forgetting, learn new knowledge, and achieve overall segmentation performance.

3) Implementation Details: Our method consists of two stages: agent training and segmentation model training. In the first stage, the agent network is trained to develop the selection policy (Section IV). While in the second stage, this trained policy is deployed for memory sample selection to train a segmentation model in a continual learning manner (Section V). For the segmentation model training stage, we follow previous works by adopting SGD as the optimizer, with a momentum value of 0.9 and an initial learning rate of 0.01 using the 'poly' learning rate decay schedule. As discussed in Section V-C, training at each continual stage includes two phases: a conventional training phase and a finetuning phase. For each continual stage on Pascal VOC 2012, the model is trained for 30 epochs during the first conventional training phase and for 5 epochs during the subsequent finetuning phase. For ADE20K, the model is trained for 60 epochs during the first conventional training phase and for 10 epochs during the subsequent finetuning phase. The batch size is set to 24 for both datasets. Following [10], the memory length $|\mathcal{M}|$ is 100 for Pascal VOC 2012 and 300 for ADE20K. We also discuss the model's performance when using different memory lengths $|\mathcal{M}|$ in Section VI-D1. The number of superpixels M in the multi-structure graph is set to 5, ϵ in Eq.5 is set to 0.1, and the number N_e of sub-experts $\boldsymbol{e}_i^n$ for each expert $\boldsymbol{E}_i$ in Section V-B is set to 4. Following most previous CNN-based CSS methods, we use Deeplabv3 with an ImageNet-pretrained ResNet101 (stride=16) as the backbone for our segmentation model. In Section VI-B2, we also extend our experiments to transformer-based segmentation methods. For the agent training stage, since this process is conducted offline, we can use a shallower segmentation network and a smaller dataset to reduce computational costs. Specifically, we use DeepLabv3 with a ResNet18 backbone as the segmentation model. The training epochs Y in Alg.1 are set to 1000. We randomly partition 10% of the entire dataset into the training set and reserve the remaining data for the reward set. For each continual stage, the network is trained for 5 epochs on Pascal VOC 2012 and 8 epochs on ADE20K. The segmentation network is optimized using SGD with an initial learning rate of 0.01, and the agent network is optimized using Momentum with a learning rate of 0.1.

B. Main Results

1) Comparison with the State-of-the-arts: We compare the performance of our method with other state-of-the-art CSS approaches on Pascal VOC 2012 and ADE20K. The results for these two datasets are presented in Table I and Table II, respectively, where the first row (Joint) represents the performance of training all classes $\mathcal{C}_{1:T}$ together without using continual learning. Compared to other methods, the conference version of our approach, denoted as the replay-based pipeline (RP) with automatic sample selection, and its extended version in this paper, denoted as replay master (RPMaster), achieve the best results across all protocols. For example, on the overlapped setting of Pascal VOC 2012, our RP achieves mIoU of 71.94% on the 15-1(6 stages) protocol, improving upon the previous best result by 9.80%. Benefiting from the newly proposed method for effectively utilizing memory samples for replay, the extended version in this paper, RPMaster, achieves even better performance, with an improvement of 1.91% on this protocol. It is important to note that when the number of continual learning stages is small, the performance gains from our RPMaster against RP are relatively modest. This is because, in such settings, the degree of forgetting previous classes is relatively limited, making the improvements introduced by our method appear less significant. However, as the number of stages increases, the problem of catastrophic forgetting becomes more severe. In these more challenging scenarios, the advantage of our approach becomes significantly more evident, as it leverages memory samples more effectively to enable better replay training. For instance, RPMaster outperforms RP by 1.91% in the 15-1 (6 stages) protocol — substantially higher than the gain of 0.50% observed in the 15-5 (2 stages) protocol. Compared to Pascal VOC 2012, ADE20K is a more challenging dataset due to its larger number of classes and images with more complex scenes. On this dataset, our method also achieves superior performance that surpasses previous results significantly. We also report and compare the results under the disjoint setting for Pascal VOC 2012 in Table I. In this setting, we cannot obtain pseudo-labels for $\mathcal{C}_{1:t-1}$ from the training samples of $\mathcal{D}_t$, thus making the catastrophic forgetting problem more severe. Under this challenging condition, the importance of information from previous classes introduced by the replay samples becomes even more crucial, thus the advantages of our proposed innovative and effective replay mechanism are more pronounced. These results demonstrate the high effectiveness of our method under diverse settings, highlighting its high robustness and excellent generalization.

2) Extension to Transformer-based Segmentation Models: Although most previous CSS methods typically use the CNN-based Deeplab as the segmentation model, the success of transformers in various computer vision fields has inspired more recent CSS works to design continual learning methods for transformer-based segmentation models. We notice that previous transformer-based CSS methods lack a unified paradigm, with different transformer models used by different methods. For example, Segmenter [69] is used by Incrementer [65] and MBS [58], and Mask2Former [19] is used by CoMFormer [8] and CoMasTRe [28]. To ensure a fair comparison and evaluate the effectiveness of our approach more comprehensively, we implement our method on both of these two transformer models (Segmenter and Mask2Former) and make a comparison with other methods. The results presented in Table III show that our method consistently outperforms existing approaches across various protocols and models. This highlights the high effectiveness and superiority of our approach when applied to transformer-based segmentation methods, demonstrating its robust generalization capabilities and substantial practical value across different model architectures.

3) Combining with Other CSS Methods: Our method is not only an independent pipeline but can also serve as a plug-and-play improvement strategy that can be used in conjunction with other CSS methods to further enhance their performance. To demonstrate this, we combine our method with three other CSS approaches—MIB [9], RBC [101], and PLOP [24]—and

TABLE I
COMPARISON RESULTS ON PASCAL VOC 2012 IN BOTH OVERLAPPED AND DISJOINT SETTINGS

Method	19-1(2 stages)						15-5(2 stages)						15-1(6 stages)					
	Overlapped			Disjoint			Overlapped			Disjoint			Overlapped			Disjoint		
	0-19	20	all	0-19	20	all	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all
Joint	78.40	78.82	78.42	78.40	78.82	78.42	79.95	73.51	78.42	79.95	73.51	78.42	79.95	73.51	78.42	79.95	73.51	78.42
EWC [40]	26.90	14.00	26.30	23.20	16.00	22.90	24.30	35.50	27.10	26.70	37.70	29.40	0.30	4.30	1.30	0.30	4.30	1.30
ILT[54]	67.75	10.88	65.05	69.10	16.40	66.40	67.08	39.23	60.45	63.20	39.50	57.30	8.75	7.99	8.56	3.70	5.70	4.20
MiB [9]	71.43	23.59	69.15	69.60	25.60	67.40	76.37	49.97	70.08	71.80	43.30	64.70	34.22	13.50	29.29	46.20	12.90	37.90
REMINDER [59]	76.48	32.34	74.38	-	-	-	76.11	50.74	70.07	-	-	-	68.30	27.23	58.52	-	-	-
SDR [55]	69.10	32.60	67.40	69.90	37.30	68.40	75.40	52.60	69.90	73.50	47.30	67.20	44.70	21.80	39.20	59.20	12.90	48.10
UCD [86]	71.40	47.30	70.00	73.40	33.70	71.50	77.50	53.10	71.30	71.90	49.50	66.20	49.00	19.50	41.90	53.10	13.00	42.90
RBC [101]	77.26	55.60	76.23	76.43	45.79	75.01	76.59	52.78	70.92	75.12	49.71	69.89	69.54	38.44	62.14	61.68	19.52	51.60
PLOP [24]	75.35	37.35	73.54	75.37	38.89	73.64	75.73	51.71	70.09	71.00	42.82	64.29	65.12	21.11	54.64	57.86	13.67	46.48
RCIL [96]	-	-	-	-	-	-	78.80	52.00	72.40	75.00	42.80	67.30	70.60	23.70	59.40	66.10	18.20	54.70
SPPA [49]	76.50	36.20	74.60	75.50	38.00	73.70	78.10	52.90	72.10	75.30	48.70	69.00	66.20	23.30	56.00	59.60	15.60	49.10
LGKD [87]	76.50	37.50	75.50	75.39	39.55	73.68	77.60	54.30	72.70	75.38	49.46	69.21	69.00	29.10	60.50	62.45	19.03	52.11
IDEC [99]	76.92	36.98	75.02	72.78	36.50	71.05	78.01	51.84	71.78	74.95	48.39	68.63	76.96	36.48	67.32	70.02	24.88	59.27
BARM [95]	**78.20**	42.20	76.40	76.75	45.10	75.24	77.43	53.03	71.62	75.09	50.11	69.14	77.60	45.90	70.00	71.78	26.65	61.03
NeST [84]	77.00	49.10	75.70	74.16	50.23	73.02	77.60	55.80	72.40	76.02	49.89	69.80	72.20	33.70	63.10	64.34	18.35	53.40
LAG [92]	76.71	45.29	75.21	73.32	47.98	72.11	77.33	51.76	71.24	76.25	49.59	69.90	75.00	37.52	66.08	70.08	25.73	59.52
RP [107]	77.86	56.75	76.85	77.10	58.92	76.23	79.31	55.88	73.73	77.18	53.04	71.43	78.54	50.82	71.94	75.49	44.67	68.15
RPMaster	78.02	**57.78**	**77.06**	**77.84**	**59.25**	**76.95**	**79.68**	**56.80**	**74.23**	**78.10**	**53.88**	**72.33**	**79.16**	**56.86**	**73.85**	**77.16**	**52.95**	**71.40**

TABLE II
COMPARISON RESULTS ON ADE20K IN OVERLAPPED SETTING

Method	100-50(2 stages)			100-10(6 stages)			100-5(11 stages)			50-50(3 stages)		
	1-100	101-150	all	1-100	101-150	all	1-100	101-150	all	1-50	51-150	all
Joint	45.28	27.95	39.54	45.28	27.95	39.54	45.28	27.95	39.54	51.56	33.41	39.54
MiB[9]	40.52	17.17	32.79	38.21	11.12	29.24	36.01	5.66	25.96	45.57	21.01	29.31
SDR [55]	37.40	24.80	33.20	28.90	7.40	21.70	-	-	-	40.90	23.80	29.50
PLOP[24]	41.87	14.89	32.94	40.48	13.61	31.59	39.10	7.80	28.80	48.83	20.99	30.40
REMINDER[59]	41.55	19.16	34.14	38.96	21.28	33.11	-	-	-	38.96	21.28	33.11
RCIL [96]	42.30	18.80	34.50	39.30	17.60	32.00	38.50	11.50	29.60	48.30	25.00	32.50
SPPA [49]	42.90	19.90	35.20	41.00	12.50	31.50	-	-	-	49.80	23.90	32.50
EWF [81]	41.20	21.30	34.60	41.50	16.34	33.20	41.40	13.40	32.10	-	-	-
LGKD [87]	43.30	25.10	37.20	42.20	20.40	34.90	40.93	18.95	33.60	49.10	27.20	34.40
IDEC [99]	42.01	18.22	34.08	40.25	17.62	32.71	39.23	14.55	31.00	47.42	25.96	33.11
BARM [95]	42.00	23.00	35.70	41.10	23.10	35.20	40.50	21.20	34.10	47.90	26.50	33.70
LAG [92]	41.64	19.73	34.34	41.00	18.69	33.56	39.96	17.22	32.38	47.69	26.12	33.31
RP [107]	44.01	25.32	37.78	43.88	25.14	37.67	43.35	18.53	35.13	50.13	25.72	33.96
RPMaster	**44.55**	**26.20**	**38.43**	**44.30**	**26.48**	**38.36**	**43.91**	**22.75**	**36.87**	**50.87**	**27.96**	**35.60**

TABLE III
COMPARISON RESULTS BASED ON TRANSFORMER-BASED SEGMENTATION MODELS ON PASCAL VOC 2012 IN OVERLAPPED SETTING.

Model	Method	19-1(2 stages)			15-1(6 stages)		
		0-19	20	all	0-15	16-20	all
Segmenter	Joint	84.34	82.05	84.23	85.20	81.13	84.23
	MBS [58]	83.30	72.00	82.76	82.60	72.20	80.06
	Incrememter [65]	82.54	60.95	82.14	79.60	59.56	75.55
	Ours	**84.09**	**73.07**	**83.57**	**84.12**	**72.80**	**81.42**
Mask2Former	Joint	79.90	73.84	79.61	82.45	70.52	79.61
	CoMFormer [8]	75.35	24.06	72.91	48.97	23.28	48.18
	CoMasTRe [28]	75.13	69.51	74.86	69.77	43.62	63.54
	Ours	**78.91**	**70.16**	**78.49**	**79.04**	**62.65**	**75.14**

TABLE IV
THE PERFORMANCE OF COMBINING OUR METHOD WITH OTHER CSS METHODS ON PASCAL VOC 2012 IN OVERLAPPED SETTING.

Method	19-1(2 stages)			15-1(6 stages)		
	0-19	20	all	0-15	16-20	all
Joint	78.40	78.82	78.42	79.95	73.51	78.42
MIB [9]	71.43	23.59	69.15	34.22	13.50	29.29
MIB + Ours	**78.05**	**58.10**	**77.10**	**79.23**	**57.01**	**73.94**
RBC [101]	77.26	55.60	76.23	69.54	38.44	62.14
RBC + Ours	**78.30**	**59.16**	**77.39**	**79.74**	**59.88**	**75.01**
PLOP [24]	75.35	37.35	73.54	65.12	21.11	54.64
PLOP + Ours	**78.26**	**57.86**	**77.29**	**79.62**	**57.25**	**74.29**

evaluate the performance improvement on the overlapped setting of Pascal VOC 2012. Specifically, on top of these existing methods, we further train an agent using the approach mentioned in Section IV and employ it to select and enhance memory samples for replay training with the way described in Section V. As shown in Table IV, this modification, which incorporates our method, improves the original MIB, RBC and PLOP by 44.65%, 12.87%, and 19.65%, respectively on the 15-1(6 stages) protocol. This significant enhancement demonstrates the generality and versatility of our approach. It is worth noting that none of these three methods originally utilizes a replay mechanism, yet the introduction of just 1% of all samples for replay can result in significant performance improvements. This indicates the high effectiveness of the replay mechanism, particularly the novel replay pipeline we

TABLE V
COMPARISON WITH OTHER SAMPLE SELECTION STRATEGIES ON PASCAL VOC 2012 IN OVERLAPPED SETTING.

Selection Strategy	19-1(2 stages) 0-19	20	all	15-1(6 stages) 0-15	16-20	all
Random Selection	73.80	56.20	72.96	74.96	44.81	67.78
iCaRL [63]	74.53	56.26	73.66	75.71	44.86	68.36
Rainbow [5]	74.64	56.27	73.76	75.88	44.90	68.50
CBES [85]	75.22	56.19	74.31	76.41	45.55	69.06
SSUL [10]	74.95	56.40	74.07	76.19	44.97	68.76
NHS	75.62	56.80	74.72	76.73	45.99	69.41
Ours (w/o Enhancement)	**77.32**	**57.69**	**76.39**	**78.01**	**49.87**	**71.31**

NHS denotes a newly-designed hand-crafted strategy using the same factors as our method (see Appendix for details). The results for random selection are derived from the average of five repetitive experiments.

TABLE VI
EFFECTIVENESS OF DIFFERENT COMPONENTS.

Method	all
Ours	**73.85**
Ours w/o ASS	67.78
Ours w/o EMU	71.94

ASS: automatic sample selection (Sec.IV). EMU: effective memory utilization (Sec.V).

TABLE VII
EFFECTIVENESS OF SELECTION AND ENHANCEMENT.

Method	all
Ours	**73.85**
Ours w/o Enh	71.31
Ours w/o Sel & Enh	67.78

Sel: sample selection. Enh: sample enhancement.

TABLE VIII
EFFECTIVENESS OF DIFFERENT COMPONENTS IN THE STATE REPRESENTATION.

Method	0-15	16-20	all
Ours	**79.16**	**56.86**	**73.85**
Ours w/o sample diversity div	74.77	50.74	69.05
Ours w/o accuracy I	77.11	53.76	71.55
Ours w/o forgetfulness g	77.34	54.02	71.79
Ours w/o $\{I, g\}$	76.27	52.39	70.58

propose in this paper, in enhancing model performance within the continual learning segmentation tasks.

4) Comparison with Other Sample Selection Strategies: To evaluate the effectiveness of our RL-driven automatic sample selection mechanism, we conduct comparative analyses against other selection methods within the CSS task. These experiments are performed using the Pascal VOC 2012 dataset under the 19-1(2 stages) and 15-1(6 stages) overlapped settings, with the results presented in Table V. The compared methods include three types: 1) the random selection strategy; 2) the previously-proposed hand-crafted strategies including iCaRL [63], Rainbow [5], CBES [85] and SSUL [10], where iCaRL and Rainbow are diversity-based selection criteria, and CBES and SSUL are class-balanced sample selection strategies specially designed for CSS. Moreover, to further validate the effectiveness of our automatic learning mechanism, we introduce and compare with a novel hand-crafted strategy (NHS) that utilizes the same factors as our RL approach—sample diversity and class performance. This strategy, detailed in the Appendix, is based on our analysis of the learned policy as described in Section VI-D2. For a fair evaluation, all methods, including ours, utilize the same replay training approach as described in Section V. On the 'all' metric of 15-1(6 stages) protocol, random selection achieves 67.78% mIoU. Employing heuristic rules for intelligent sample selection, iCaRL, Rainbow, CBES, and SSUL achieve mIoUs of 68.36%, 68.50%, 69.06%, and 68.76%, respectively. NHS further enhances this performance to 69.41% by considering additional factors and their complex interrelationships. To maintain fairness in comparison, we report the performance of our method without the sample enhancement operation, which achieves 71.31% mIoU. This result not only outperforms the previously proposed iCaRL, Rainbow, CBES, and SSUL, highlighting the higher effectiveness of our novel selection approach; but also surpasses NHS that uses the same set of factors as ours, thus demonstrating the significant advantages of the reward-driven automatic policy learning mechanism over the hand-crafted strategies.

C. Ablation Study

In this section, we conduct ablation study experiments to verify the effectiveness of different components and design choices in our method. The experiments are conducted on Pascal VOC 2012 under the 15-1 (6 stages) overlapped setting.

1) Effectiveness of Different Components: Our method consists of two main components: automatic sample selection (ASS, Section IV) and effective memory utilization (EMU, Section V). As shown in Table VI, when the proposed selection method is not used and samples are selected randomly, or when the proposed effective utilization method is not applied and conventional replay training is used instead, the model's performance will significantly decrease. These results demonstrate that both components of our pipeline contribute significantly to the high-performance continual segmentation.

2) Effectiveness of Selection-enhancement Dual-stage Action: As detailed in Section IV-D, the action in our method consists of two stages: sample selection and sample enhancement. We conduct experiments to verify the effectiveness of these two stages, and the results are presented in Table VII. Our method, which includes both sample selection and enhancement, achieves 73.85% mIoU on the 'all' metric. Removing the enhancement operation reduces performance to 71.31%. Further removing both the enhancement and selection procedures and employing a randomly filled memory decreases performance to 67.78%, which is 6.07% lower than our full method. These results indicate that both the selection and enhancement operations in our method can effectively boost CSS performance, demonstrating the soundness and importance of our designs for the replay pipeline.

3) Ablation Study of the State Representation Components: We then validate the different components of the designed state representations and the results are presented in Table VIII. As introduced in Section IV-B, the state representation in our method consists of three components: 1) sample diversity div; 2) accuracy I and 3) forgetfulness g, with the latter two constituting the class performance feature. As shown in Table VIII, removing any of these components can lead to a significant decrease in mIoU scores, demonstrating their effectiveness and importance in making a wiser selection decision

TABLE IX
EFFECTIVENESS AND EFFICIENCY OF SIMILARITY COMPUTATION.

Method	0-15	16-20	all	KFLOPs
Graph-based Method (Ours)	79.16	56.86	73.85	221.7
Prototype-level Method	76.79	53.27	71.19	**0.8**
Pixel-level Method	**79.36**	**57.43**	74.14	9837.3
Ours w/o d_{se} in graph	77.42	53.80	71.80	186.6
Ours w/o d_{sp} in graph	77.95	54.26	72.31	221.6

KFLOPs here refers to the average computational cost required to calculate the similarity between every two images (including the computation for superpixel construction in our graph-based method).

TABLE X
COMPUTATION OF DIVERSITY div AND FORGETFULNESS g IN STATE REPRESENTATIONS.

Method	0-15	16-20	all	FLOPs
Sampling 10% of images as $\mathcal{S}_c$	79.16	56.86	73.85	**27.8M**
All images as $\mathcal{S}_c$	**79.35**	**57.12**	**74.06**	277.8M
Sampling 10% from $\mathcal{S}_c$ as $\hat{\mathcal{S}}_c$	79.16	56.86	73.85	**0.7G**
All images from $\mathcal{S}_c$ as $\hat{\mathcal{S}}_c$	**79.30**	**57.49**	**74.11**	73.9G

FLOPS here refers to the average computational cost required to calculate div for each image and g for each classs.

TABLE XI
ABLATION STUDY OF EFFECTIVE MEMORY UTILIZATION.

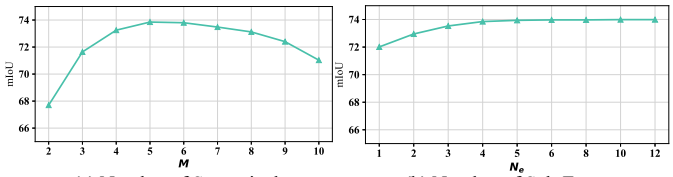
Method	0-15	16-20	all
Ours	**79.16**	**56.86**	**73.85**
Ours w/o experts $\boldsymbol{E}$ (Sec.V-B)	77.83	53.55	72.05
Ours w/o 1st phase of the dual-phase training (Sec.V-C)	71.17	46.93	65.40
Ours w/o 2nd phase of the dual-phase training (Sec.V-C)	78.38	55.66	72.97
Ours w/o shared information utilization	78.41	55.14	72.87
Ours w/o τ in Eq.8	78.69	55.31	73.12
Ours w/ all samples from $\mathcal{M} \cup \mathcal{D}_t$ for finetuning	78.50	55.78	73.09
Ours w/o $\mathcal{L}_{div}$ in Eq.10	78.42	55.79	73.03
Ours w/ all parameters to be updated in finetuning	78.40	55.68	72.99

that contributes to the higher segmentation performance.

4) Effectiveness and Efficiency of Similarity Computation: As illustrated in Section IV-C, to compute the sample diversity and forgetfulness in our state representations more effectively and efficiently, we propose a novel multi-structure graph to calculate inter-sample similarity. To demonstrate the advantages of our method, we compare it with two conventional similarity measurement approaches—the prototype-level method and the pixel-level method—and present the results in Table IX. Specifically, the prototype-level method first employs masked average pooling to obtain the representation vector of a class c within an image. Then, the cosine distance between the representation vectors of two images is calculated as their inter-sample similarity for class c. The pixel-level method calculates and averages the similarity between every pair of pixels belonging to class c in two images. More concretely, let $\{p_1^i\}_{i=1}^{N_1}$ and $\{p_2^j\}_{j=1}^{N_2}$ represent the sets of backbone features for all pixels belonging to c in two images x_1 and x_2, respectively. The pixel-level method computes their similarity by: $\frac{1}{N_1 N_2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\mathrm{Cos}(p_1^i, p_2^j)$, where Cos refers to the cosine distance. As shown in Table IX, the prototype-level method requires the least computation since it only needs to calculate the cosine distance once when measuring the similarity between two images. However, as discussed in detail in Section IV-C1, this method performs poorly due to the lack of consideration for local details and structured information. The pixel-level method retains local information and therefore achieves better performance, but the requirement to compute similarity for every pair of pixels results in a substantial computational burden. Our method successfully achieves a balance between effectiveness and computational efficiency by using the proposed multi-structure graph, which retains the spatial and structure information while reducing the computational load by condensing the feature maps into fewer superpixels. As a result, our method requires significantly less computation than the pixel-level method while achieving nearly comparable performance. We further evaluate the designs within the proposed multi-structure graph. Specifically, as detailed in Section IV-C2, the edges of the graph are calculated as the combination of semantic distance d_{se} and spatial distance d_{sp}. As shown in Table IX, removing either distance could negatively impact the model performance, which demonstrates the effectiveness and importance of both components.

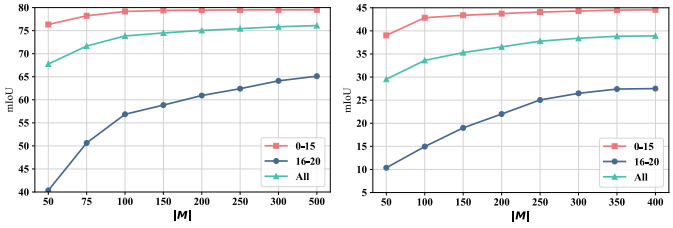
5) Computation of diversity div and forgetfulness g: As introduced in Section IV-C4, we adopt two strategies to further reduce the computational load of calculating diversity div and forgetfulness g in our state representations at stage t: 1) We randomly sample 10% of all images within each class c in $\mathcal{C}_t$ to construct its support set $\mathcal{S}_c$. 2) We select the top 10% of samples in $\mathcal{S}_c$ with the lowest diversity scores div to construct the representative set $\hat{\mathcal{S}}_c$. Please refer to Section IV-C4 for more details of these methods. As presented in Table X, if we use all samples of class c to construct $\mathcal{S}_c$, or all samples from $\mathcal{S}_c$ to construct $\hat{\mathcal{S}}_c$, the model's performance can be slightly improved due to the more accurate calculation of div and g from a larger number of samples. However, the significantly increased computational load would reduce the method's efficiency. In contrast, our approach largely reduces the computation with only a minimal sacrifice in accuracy, thus achieving a better balance between model effectiveness and computation efficiency.

6) Ablation Study of Effective Memory Utilization: In Section V, we propose a novel method to utilize memory samples for replay to effectively overcome the issue of catastrophic forgetting. We further evaluate the effectiveness of different components within this method and present the results in Table XI. Specifically, our approach introduces a new expert network with shared information utilization (Section V-B) and training method (Section V-C) to achieve better memory utilization. When we remove either of these components, i.e., by eliminating the experts $\boldsymbol{E}$ for each class, or by not using the dual-phase training and instead performing only one of the two stages, the segmentation performance decreases by 1.80%, 8.45% and 0.88%, respectively. We further evaluate the effectiveness of the designs within our expert mechanism and dual-phase training method. Specifically, for the expert mechanism, we conduct the following experiments: 1) restricting the segmentation process of each $\mathcal{C}_i$ to only use features extracted by its corresponding expert $\boldsymbol{E}_i$ without shared information utilization, and 2) not using the random



(a) Number of Superpixels (b) Number of Sub-Experts

Fig. 4. Performance on Pascal VOC 2012 15-1(6 stages) overlapped setting when using different values for the number of superpixels M (a) and the number of sub-experts N_e (b).



(a) Pascal VOC 2012 15-1(6stages) (b) ADE20K 100-10(6stages)

Fig. 5. Performance on Pascal VOC 2012 (a) and ADE20K (b) when setting the capacity $|\mathcal{M}|$ of memory $\mathcal{M}$ to different values.

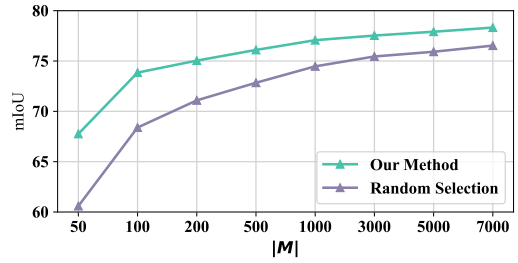


Fig. 6. Performance comparison on the Pascal VOC 2012 15-1(6 stages) setting when using our method and random method for replay sample selection. Results for different memory capacities are reported.

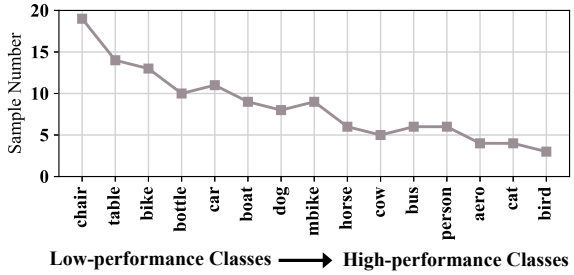


Fig. 7. The numbers of selected samples for different classes. The horizontal axis from left to right represents classes from poor to good performance.

interruption strategy implemented by τ in Eq.8, and instead directly applying Eq.7 during training. For the training method, we 1) do not use the balanced $\mathcal{M} \cup \hat{\mathcal{D}}_t$ but use all samples $\mathcal{M} \cup \mathcal{D}_t$ for finetuning, 2) remove the diversity loss $\mathcal{L}_{div}$ from the loss functions in Eq.10, and 3) allow all model parameters to be updated during the finetuning phase. We find that either of these modifications to our original method would result in a decrease in performance. These results demonstrate the effectiveness and importance of our methods and designs.

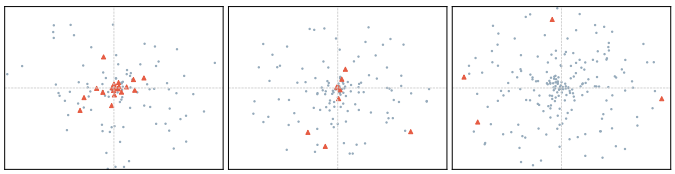
7) Ablation Study of Hyperparameters: We further conduct experiments to determine the best choices for the hyperparameters in our method, including the number of superpixels M in the multi-structure graph and the number N_e of sub-experts $\boldsymbol{e}_i^n$ for each expert $\boldsymbol{E}_i$ as discussed in Section V-B, with the results presented in Figure 4(a) and Figure 4(b), respectively. As observed from Figure 4(a), the performance remains stable when M is larger than 3 and less than 8, while an overly large M may lead to over-segmentation, which could negatively affect segmentation performance. For the experiments on N_e shown in Figure 4(b), we find that a larger N_e can improve performance since more diverse features are extracted and utilized, but it will also increase the number of parameters. When $N_e > 5$, the model's performance approaches saturation, so further increasing N_e only yields limited improvement. In conclusion, our method is not sensitive to the setting of hyperparameters, as the model can consistently achieve stable and excellent performance when $3 < M < 8$ and $N_e > 3$.

D. Discussion and Analysis

1) Influence of Memory Capacity: It is intuitive that the capacity $|\mathcal{M}|$ of the memory $\mathcal{M}$ can significantly impact the model's performance. As shown by the results presented in Figure 5, a larger capacity for $\mathcal{M}$ allows more replay samples to be involved in the training of the next stage, thereby alleviating the catastrophic forgetting issue better and achieving higher performance. In this paper, we follow the setup of [10] by setting $|\mathcal{M}|$ to 100 for Pascal VOC 2012 and 300 for ADE20K. The results indicate that even when using only 1% of the total dataset ($|\mathcal{M}|$=100 and the length of the entire training set being 10582 for Pascal VOC 2012) as memory samples in this setting, the model can already achieve excellent performance. This highlights the high efficiency of our replay pipeline benefiting from our carefully designed sample selection mechanism and utilization method. Furthermore, we conduct an additional comparison between our approach and the random method for replay sample selection under varying memory capacities $|\mathcal{M}|$. As shown in Figure 6, our method exhibits a substantial advantage when the memory size $|\mathcal{M}|$ is small. Although the performance gain from our sample selection method becomes less pronounced as the memory size increases, it remains very significant. For example, when $|\mathcal{M}| = 7000$, which corresponds to storing over 60% of the training samples, our method still outperforms the random selection strategy by 1.53%. These results demonstrate that our method can consistently achieve better performance than random selection across a wide range of memory capacities, highlighting its high effectiveness and general applicability.

2) Analysis of the Learned Policy: We further analyze the learned sample selection policy to provide deeper insights into how our method works, and we found the following rules:

(a) Low-performance classes require more replay samples. As shown in Figure 7, on the 15-1 (6 stages) protocol of the Pascal VOC 2012 dataset, after completing the memory sample selection at the end of the first stage, we count the number of selected samples for different classes with varying performance levels. Specifically, from left to right,



(a) Chair (b) Boat (c) Cat

Fig. 8. Visualization of the diversity for the selected samples of three classes including 'chair', 'boat' and 'cat'. The red triangles represent the selected samples and the gray dots denote other samples that are not selected. Triangles or dots closer to the center represent samples with the lower diversity. (Best viewed in color)

the horizontal axis of Figure 7 represents different classes in order of increasing performance, where the performance of a class c is measured by summing its accuracy factor I_c and forgetfulness factor g_c as illustrated in Section IV-B. It is observed that the number of selected samples of a class is negatively correlated with its performance. This is because low-performance classes are either less accurate or more prone to forgetting, so more samples are required for replay to further improve segmentation capabilities for these classes or to mitigate their more severe issues of catastrophic forgetting.

(b) Classes with different forgetfulness require different kinds of samples. We further investigate the correlation between class forgetfulness and sample diversity in the learned selection policy and discover some interesting patterns. Specifically, we select three representative classes: 'chair', 'boat', and 'cat', and visualize the diversity scores of their selected samples. Among these classes, 'chair' is a difficult class with a low forgetfulness score g that indicates it is more prone to be forgotten, while 'cat' represents an easier class with a higher score, and 'boat' exhibits a medium score. The results for these classes are shown in Figure 8, where the red triangles represent the selected samples, and the gray dots denote other non-selected ones. Triangles or dots closer to the center represent samples with lower diversity scores that indicate they are more similar to others within the same class. Our results reveal distinct selection strategies across these classes. For the hard class 'chair', we find that most red triangles are distributed near the center, indicating that common samples with low diversity are selected. In contrast, the 'cat' class, which is less susceptible to forgetting, exhibits a preference for high-diversity samples. The 'boat' class demonstrates an intermediate pattern, selecting both common and diverse samples for replay. These patterns could be explained by the differing types of replay samples required by different classes with varying levels of forgetfulness. Specifically, for hard classes where catastrophic forgetting is more severe, most samples—including both high-diversity novel ones and low-diversity common ones—could be forgotten after the model trains on new classes. Therefore, using more common and representative samples helps to learn a more general classification space that covers most samples. In contrast, for easy classes with relatively minor catastrophic forgetting issues, the common samples are more likely to be still remembered in the next stage, while the high-diversity samples, which are further from the class's distribution center, are more likely to be forgotten. Thus, replaying with high-diversity samples can be more effective for such classes.

TABLE XII
COMPARISON OF COMPUTATIONAL COST BETWEEN RP AND RPMASTER.

Method	Pascal 15-1(6 stages)			ADE 100-5(11 stages)		
	hour/stage	GFLOPs	mIoU	hour/stage	GFLOPs	mIoU
RP [107]	**0.53**	**68.17**	71.94	**2.24**	**68.18**	35.13
RPMaster	0.64	69.10	**73.85**	2.76	70.26	**36.87**

"hour/stage" denotes the training time for each continual stage on 2 V100 GPUs.

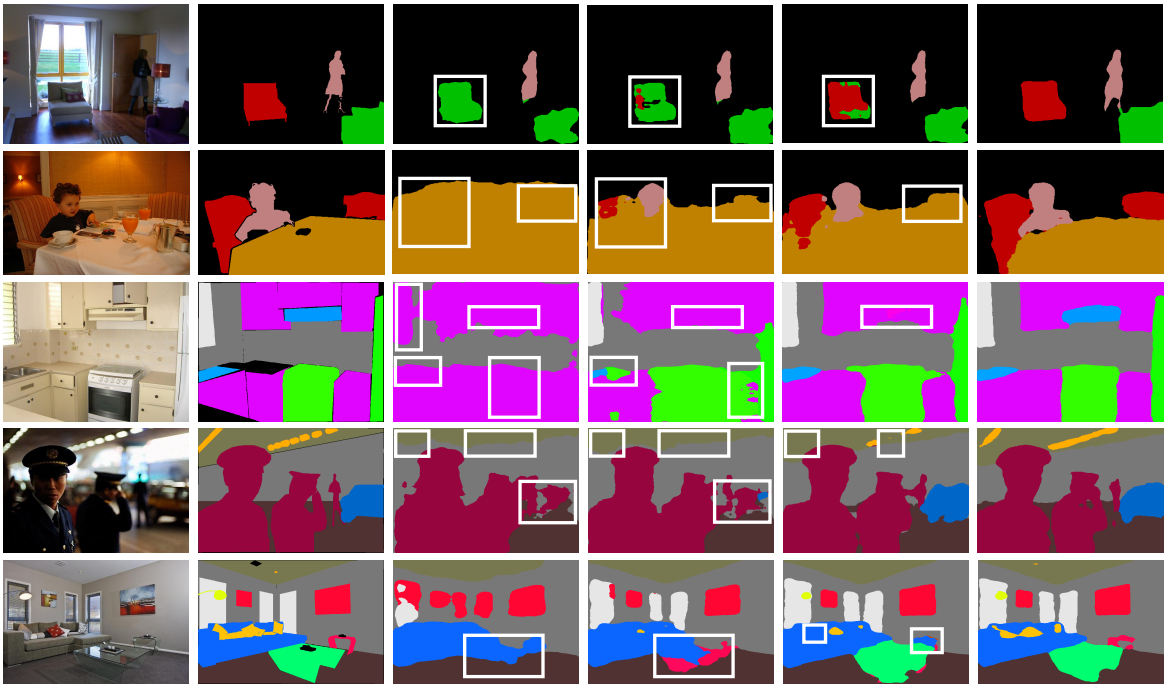
TABLE XIII
CROSS-DATASET DEPLOYMENT EXPERIMENTS THAT DEPLOY THE AGENT TRAINED FROM ONE DATASET TO ANOTHER DATASET

Deployment Dataset	Sample Selection Method	mIoU
Pascal VOC 2012	Random	70.17
	Agent trained from Pascal VOC 2012	**74.23**
	Agent trained from ADE20K	74.05
ADE20K	Random	34.11
	Agent trained from ADE20K	**38.43**
	Agent trained from Pascal VOC 2012	37.89

E. Discussion of Complexity and Generalization

1) Efficiency Enhancement for Agent Training: In our method, training the agent network incurs very high computational costs due to the complex policy updates, the use of multiple networks, temporal-difference (TD) learning, and experience replay mechanisms involved in our reinforcement learning framework. Specifically, as shown in Alg.1, the theoretical additional cost for agent training is $O(Y)$ higher than that of deployment. Fortunately, we find that the agent exhibits strong generalization ability, and thus, as detailed in Section VI-A3, we can train the agent using a shallower segmentation network and a smaller dataset, without needing to replicate the exact settings used during deployment. Empirically, we found that training with a ResNet-18-based segmentation model and only 10% of the full dataset is sufficient to obtain a powerful agent that generalizes effectively to deeper segmentation networks based on the ResNet-101 backbone. These simplifications enable a computation-efficient training process, with the agent training time being only about 12 hours on the 15-1 (6 stages) setting of Pascal VOC 2012.

2) Comparison of Computational Cost Between RP and PRMaster: Compared to the conference version RP [107], the extended method RPMaster in this work incurs higher computational costs due to the additional expert modules and the dual-phase training strategy, as shown in Table XII, which reports both the training time per continual stage (hour/stage) on 2 NVIDIA TITAN V100 GPUs and the GFLOPs during inference. However, in RPMaster, the additional training epochs required for each continual stage are relatively small (only 5 epochs for Pascal VOC 2012 and 10 for ADE20K), and the expert modules are very lightweight, consisting only of some standard convolutional layers. Therefore, RPMaster's increase in computational cost compared to RP is relatively minor. Considering the significant performance improvements



(a) Image (b) Ground Truth (c) PLOP (d) BARM (e) RP (f) RPMaster

Fig. 9. Visualization comparison of different CSS methods. From left to right, each row shows the image, ground truth, segmentation results obtained by PLOP [24], BARM [95], RP [107], and our RPMaster. The regions outlined by white bounding boxes indicate the erroneous areas in the segmentation results.

achieved by RPMaster, we believe this modest additional cost is well justified and entirely acceptable.

3) Cross-Dataset Generalization: We further found that the trained agent exhibits strong cross-dataset generalization, as an agent trained on one dataset can be effectively deployed on others. As shown in Table XIII, using the agent trained on Pascal VOC 2012 to deploy on the 100-50 (2 stages) setting of ADE 20K achieves 37.89% mIoU, and using the agent trained on ADE 20K to deploy on the 15-5 (2 stages) setting of Pascal VOC 2012 achieves 74.05% mIoU. In both cases, the performance significantly exceeds that of using randomly selected samples and is only slightly below the result of using an agent trained on the same dataset as deployment. These results demonstrate the high generalization capability of our method. In practical applications, the agent only needs to be trained once and can then be used for several different CSS tasks without the additional computation cost for agent retraining. This significantly reduces the computational demands for completing CSS training across multiple datasets, showcasing the high practical value of our method in real-world scenarios.

F. Visualization of Segmentation Results

In Figure 9, we present the visualization of segmentation results obtained from different CSS methods. The results on both the Pascal VOC 2012 (1st–2nd rows) and ADE20K (3rd–5th rows) are covered. As shown in the figure, previous methods like PLOP [24] and BARM [95] yield suboptimal results with several erroneous regions. In contrast, our conference version RP [107], which employs an RL-based automatic sample selection strategy, significantly improves segmentation quality by selecting more informative samples for replay. Building upon RP, the extended RPMaster framework proposed in this paper introduces an effective memory utilization strategy to further alleviate class imbalance issues. As a result, RPMaster achieves even better segmentation performance than the conference version, demonstrating the high effectiveness and substantial improvements introduced in our extended work.

VII. Conclusion

This work proposes a novel replay-based pipeline for continual semantic segmentation, which includes an automatic memory sample selection mechanism powered by a task-tailored reinforcement learning framework, and an effective approach to utilize memory samples for better replay through an expert mechanism and a dual-phase training method. Extensive experiments across multiple datasets and protocols demonstrate the high effectiveness and generalization of our method, which achieves state-of-the-art (SOTA) performance while utilizing only 1% of the total training data for replay. We consider our work as an important method that can provide valuable insights into memory selection and utilization in continual semantic segmentation, making the replay methods in this research domain more effective and readily applicable.

References

[1] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Proc. Adv. Neural Inf. Process. Syst.*, 32, 2019.

[2] Manoj Alwani, Yang Wang, and Vashisht Madhavan. Decore: Deep compression with reinforcement learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12349–12359, 2022.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(12):2481–2495, 2017.

[4] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[5] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8218–8227, 2021.

[6] Zalán Borsos, Mojmir Mutny, and Andreas Krause. Coresets via bilevel optimization for continual learning and streaming. *Proc. Adv. Neural Inf. Process. Syst.*, 33:14879–14890, 2020.

[7] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 221–230, 2017.

[8] Fabio Cermelli, Matthieu Cord, and Arthur Douillard. Comformer: Continual learning in semantic and panoptic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3010–3020, 2023.

[9] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9233–9242, 2020.

[10] Sungmin Cha, YoungJoon Yoo, Taesup Moon, et al. Ssul: Semantic segmentation with unknown label for exemplar-based class-incremental learning. *Proc. Adv. Neural Inf. Process. Syst.*, 34:10919–10930, 2021.

[11] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4974–4983, 2019.

[12] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

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

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

[15] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Eur. Conf. Comput. Vis.*, pages 801–818, 2018.

[16] Ting Chen, Lala Li, Saurabh Saxena, Geoffrey Hinton, and David J Fleet. A generalist framework for panoptic segmentation of images and videos. In *Int. Conf. Comput. Vis.*, pages 909–919, 2023.

[17] Tianrun Chen, Lanyun Zhu, Chaotao Deng, Runlong Cao, Yan Wang, Shangzhan Zhang, Zejian Li, Lingyun Sun, Ying Zang, and Papa Mao. Sam-adapter: Adapting segment anything in underperformed scenes. In *Int. Conf. Comput. Vis.*, pages 3367–3375, 2023.

[18] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12475–12485, 2020.

[19] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1290–1299, 2022.

[20] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Proc. Adv. Neural Inf. Process. Syst.*, 34:17864–17875, 2021.

[21] Wei Cong, Yang Cong, Yuyang Liu, and Gan Sun. Cs2k: Class-specific and class-shared knowledge guidance for incremental semantic segmentation. In *Eur. Conf. Comput. Vis.*, 2024.

[22] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016.

[23] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Proc. Adv. Neural Inf. Process. Syst.*, 26, 2013.

[24] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. Plop: Learning without forgetting for continual semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4040–4050, 2021.

[25] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *Int. J. Comput. Vis.*, 111:98–136, 2015.

[26] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3146–3154, 2019.

[27] Jia Gong, Zhipeng Fan, Qiuhong Ke, Hossein Rahmani, and Jun Liu. Meta agent teaming active learning for pose estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11079–11089, 2022.

[28] Yizheng Gong, Siyue Yu, Xiaoyang Wang, and Jimin Xiao. Continual segmentation with disentangled objectness learning and class recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3848–3857, 2024.

[29] Jiaxin Guo, Fangxun Zhong, Rong Xiong, Yunhui Liu, Yue Wang, and Yiyi Liao. A visual navigation perspective for category-level object pose estimation. *arXiv preprint arXiv:2203.13572*, 2022.

[30] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Int. Conf. Comput. Vis.*, pages 2961–2969, 2017.

[31] Hanzhe Hu, Yinbo Chen, Jiarui Xu, Shubhankar Borse, Hong Cai, Fatih Porikli, and Xiaolong Wang. Learning implicit feature alignment function for semantic segmentation. In *Eur. Conf. Comput. Vis.*, pages 487–505. Springer, 2022.

[32] Jitesh Jain, Jiachen Li, Mang Tik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. Oneformer: One transformer to rule universal image segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2989–2998, 2023.

[33] Deyi Ji, Haoran Wang, Mingyuan Tao, Jianqiang Huang, Xian-Sheng Hua, and Hongtao Lu. Structural and statistical texture knowledge distillation for semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 16876–16885, 2022.

[34] Deyi Ji, Feng Zhao, Lanyun Zhu, Wenwei Jin, Hongtao Lu, and Jieping Ye. Discrete latent perspective learning for segmentation and detection. In *Int. Conf. Mach. Learn.*, 2024.

[35] Siyu Jiao, Yunchao Wei, Yaowei Wang, Yao Zhao, and Humphrey Shi. Learning mask-aware clip representations for zero-shot segmentation. *Proc. Adv. Neural Inf. Process. Syst.*, 36:35631–35653, 2023.

[36] Xisen Jin, Arka Sadhu, Junyi Du, and Xiang Ren. Gradient-based editing of memory examples for online task-free continual learning. *arXiv preprint arXiv:2006.15294*, 2020.

[37] Lei Ke, Yu-Wing Tai, and Chi-Keung Tang. Occlusion-aware instance segmentation via bilayer network architectures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2023.

[38] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9404–9413, 2019.

[39] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Int. Conf. Comput. Vis.*, pages 4015–4026, 2023.

[40] 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.

[41] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024.

[42] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13906–13915, 2020.

[43] Gen Li, Varun Jampani, Laura Sevilla-Lara, Deqing Sun, Jonghyun Kim, and Joongkyu Kim. Adaptive prototype learning and allocation for few-shot segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8334–8343, 2021.

[44] Liulei Li, Wenguan Wang, and Yi Yang. Logicseg: Parsing visual semantics with neural logic learning and reasoning. In *Int. Conf. Comput. Vis.*, pages 4122–4133, 2023.

[45] Liulei Li, Wenguan Wang, Tianfei Zhou, Ruijie Quan, and Yi Yang. Semantic hierarchy-aware segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2023.

[46] Zhiqi Li, Wenhai Wang, Enze Xie, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, Ping Luo, and Tong Lu. Panoptic segformer: Delving deeper into panoptic segmentation with transformers. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1280–1289, 2022.

[47] Chen Liang, Wenguan Wang, Jiaxu Miao, and Yi Yang. Gmmseg: Gaussian mixture based generative semantic segmentation models. *Proc. Adv. Neural Inf. Process. Syst.*, 35:31360–31375, 2022.

[48] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7061–7070, 2023.

[49] Zihan Lin, Zilei Wang, and Yixin Zhang. Continual semantic segmen-

tation via structure preserving and projected feature alignment. In *Eur. Conf. Comput. Vis.*, pages 345–361. Springer, 2022.
[50] Zihan Lin, Zilei Wang, and Yixin Zhang. Preparing the future for continual semantic segmentation. In *Int. Conf. Comput. Vis.*, pages 11910–11920, 2023.
[51] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Rmm: Reinforced memory management for class-incremental learning. *Proc. Adv. Neural Inf. Process. Syst.*, 34:3478–3490, 2021.
[52] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3431–3440, 2015.
[53] Andrea Maracani, Umberto Michieli, Marco Toldo, and Pietro Zanuttigh. Recall: Replay-based continual learning in semantic segmentation. In *Int. Conf. Comput. Vis.*, pages 7026–7035, 2021.
[54] Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. In *Int. Conf. Comput. Vis. Worksh.*, pages 0–0, 2019.
[55] Umberto Michieli and Pietro Zanuttigh. Continual semantic segmentation via repulsion-attraction of sparse and disentangled latent representations. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1114–1124, 2021.
[56] Rohit Mohan and Abhinav Valada. Efficientps: Efficient panoptic segmentation. *Int. J. Comput. Vis.*, 129(5):1551–1579, 2021.
[57] Youngmin Oh, Donghyeon Baek, and Bumsub Ham. Alife: Adaptive logit regularizer and feature replay for incremental semantic segmentation. *Proc. Adv. Neural Inf. Process. Syst.*, 35:14516–14528, 2022.
[58] Gilhan Park, WonJun Moon, SuBeen Lee, Tae-Young Kim, and Jae-Pil Heo. Mitigating background shift in class-incremental semantic segmentation. In *Eur. Conf. Comput. Vis.*, 2024.
[59] Minh Hieu Phan, Son Lam Phung, Long Tran-Thanh, Abdesselam Bouzerdoum, et al. Class similarity weighted knowledge distillation for continual semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 16866–16875, 2022.
[60] Jie Qin, Jie Wu, Pengxiang Yan, Ming Li, Ren Yuxi, Xuefeng Xiao, Yitong Wang, Rui Wang, Shilei Wen, Xin Pan, et al. Freeseg: Unified, universal and open-vocabulary image segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 19446–19455, 2023.
[61] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PmLR, 2021.
[62] Hanoona Rasheed, Muhammad Maaz, Sahal Shaji, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M Anwer, Eric Xing, Ming-Hsuan Yang, and Fahad S Khan. Glamm: Pixel grounding large multimodal model. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13009–13018, 2024.
[63] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2001–2010, 2017.
[64] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
[65] Chao Shang, Hongliang Li, Fanman Meng, Qingbo Wu, Heqian Qiu, and Lanxiao Wang. Incrementer: Transformer for class-incremental semantic segmentation with knowledge distillation focusing on old class. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7214–7224, 2023.
[66] Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. In *Proc. AAAI Conf. Artif. Intell.*, volume 35, pages 9630–9638, 2021.
[67] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.
[68] Bharat Singh, Rajesh Kumar, and Vinay Pratap Singh. Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review*, 55(2):945–990, 2022.
[69] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Int. Conf. Comput. Vis.*, pages 7262–7272, 2021.
[70] Zhi Tian, Bowen Zhang, Hao Chen, and Chunhua Shen. Instance and panoptic segmentation using conditional convolutions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(1):669–680, 2022.
[71] Rishabh Tiwari, Krishnateja Killamsetty, Rishabh Iyer, and Pradeep Shenoy. Gcr: Gradient coreset based replay buffer selection for continual learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 99–108, 2022.
[72] Marco Toldo, Umberto Michieli, and Pietro Zanuttigh. Learning with style: Continual semantic segmentation across tasks and domains. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.
[73] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *Int. J. Comput. Vis.*, 104:154–171, 2013.
[74] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proc. AAAI Conf. Artif. Intell.*, volume 30, 2016.
[75] Huyong Wang, Huisi Wu, and Jing Qin. Incremental nuclei segmentation from histopathological images via future-class awareness and compatibility-inspired distillation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11408–11417, 2024.
[76] Haochen Wang, Xudong Zhang, Yutao Hu, Yandan Yang, Xianbin Cao, and Xiantong Zhen. Few-shot semantic segmentation with democratic attention networks. In *Eur. Conf. Comput. Vis.*, pages 730–746. Springer, 2020.
[77] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Solo: A simple framework for instance segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(11):8587–8601, 2021.
[78] Xinlong Wang, Xiaosong Zhang, Yue Cao, Wen Wang, Chunhua Shen, and Tiejun Huang. Seggpt: Towards segmenting everything in context. In *Int. Conf. Comput. Vis.*, pages 1130–1140, 2023.
[79] Yan Wang, Jian Cheng, Yixin Chen, Shuai Shao, Lanyun Zhu, Zhenzhou Wu, Tao Liu, and Haogang Zhu. Fvp: Fourier visual prompting for source-free unsupervised domain adaptation of medical image segmentation. *arXiv preprint arXiv:2304.13672*, 2023.
[80] Wei Wu, Jiawei Liu, Kecheng Zheng, Qibin Sun, and Zheng-Jun Zha. Temporal complementarity-guided reinforcement learning for image-to-video person re-identification. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7319–7328, June 2022.
[81] Jia-Wen Xiao, Chang-Bin Zhang, Jiekang Feng, Xialei Liu, Joost van de Weijer, and Ming-Ming Cheng. Endpoints weight fusion for class incremental semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7204–7213, 2023.
[82] Enze Xie, Wenhai Wang, Mingyu Ding, Ruimao Zhang, and Ping Luo. Polarmask++: Enhanced polar representation for single-shot instance segmentation and beyond. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(9):5385–5400, 2021.
[83] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Proc. Adv. Neural Inf. Process. Syst.*, 34:12077–12090, 2021.
[84] Zhengyuan Xie, Haiquan Lu, Jia-wen Xiao, Enguang Wang, Le Zhang, and Xialei Liu. Early preparation pays off: New classifier pre-tuning for class incremental semantic segmentation. In *Eur. Conf. Comput. Vis.*, pages 183–201. Springer, 2024.
[85] Shipeng Yan, Jiale Zhou, Jiangwei Xie, Songyang Zhang, and Xuming He. An em framework for online incremental learning of semantic segmentation. In *ACM Int. Conf. Multimedia*, pages 3052–3060, 2021.
[86] Guanglei Yang, Enrico Fini, Dan Xu, Paolo Rota, Mingli Ding, Moin Nabi, Xavier Alameda-Pineda, and Elisa Ricci. Uncertainty-aware contrastive distillation for incremental semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(2):2567–2581, 2022.
[87] Ze Yang, Ruibo Li, Evan Ling, Chi Zhang, Yiming Wang, Dezhao Huang, Keng Teck Ma, Minhoe Hur, and Guosheng Lin. Label-guided knowledge distillation for continual semantic segmentation on 2d images and 3d point clouds. In *Int. Conf. Comput. Vis.*, pages 18601–18612, 2023.
[88] Zongxin Yang, Yunchao Wei, and Yi Yang. Collaborative video object segmentation by multi-scale foreground-background integration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(9):4701–4712, 2021.
[89] Kaining Ying, Qing Zhong, Weian Mao, Zhenhua Wang, Hao Chen, Lin Yuanbo Wu, Yifan Liu, Chengxiang Fan, Yunzhi Zhuge, and Chunhua Shen. Ctvis: Consistent training for online video instance segmentation. In *Int. Conf. Comput. Vis.*, pages 899–908, 2023.
[90] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. *arXiv preprint arXiv:2106.01085*, 2021.
[91] Bo Yuan and Danpei Zhao. A survey on continual semantic segmentation: Theory, challenge, method and application. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.
[92] Bo Yuan, Danpei Zhao, and Zhenwei Shi. Learning at a glance: Towards interpretable data-limited continual semantic segmentation via semantic-invariance modelling. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.

[93] Haobo Yuan, Xiangtai Li, Chong Zhou, Yining Li, Kai Chen, and Chen Change Loy. Open-vocabulary sam: Segment and recognize twenty-thousand classes interactively. In *Eur. Conf. Comput. Vis.*, pages 419–437. Springer, 2024.
[94] Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learn. Syst.*, 23(8):1177–1193, 2012.
[95] Anqi Zhang and Guangyu Gao. Background adaptation with residual modeling for exemplar-free class-incremental semantic segmentation. In *Eur. Conf. Comput. Vis.*, 2024.
[96] Chang-Bin Zhang, Jia-Wen Xiao, Xialei Liu, Ying-Cong Chen, and Ming-Ming Cheng. Representation compensation networks for continual semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7053–7064, 2022.
[97] Yurong Zhang, Liulei Li, Wenguan Wang, Rong Xie, Li Song, and Wenjun Zhang. Boosting video object segmentation via space-time correspondence learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2246–2256, 2023.
[98] Zekang Zhang, Guangyu Gao, Jianbo Jiao, Chi Harold Liu, and Yunchao Wei. Coinseg: Contrast inter-and intra-class representations for incremental segmentation. In *Int. Conf. Comput. Vis.*, pages 843–853, 2023.
[99] Danpei Zhao, Bo Yuan, and Zhenwei Shi. Inherit with distillation and evolve with contrast: Exploring class incremental semantic segmentation without exemplar memory. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(10):11932–11947, 2023.
[100] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2881–2890, 2017.
[101] Hanbin Zhao, Fengyu Yang, Xinghe Fu, and Xi Li. Rbc: Rectifying the biased context in continual semantic segmentation. In *Eur. Conf. Comput. Vis.*, pages 55–72. Springer, 2022.
[102] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 633–641, 2017.
[103] Tianfei Zhou and Wenguan Wang. Cross-image pixel contrasting for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.
[104] Tianfei Zhou and Wenguan Wang. Prototype-based semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.
[105] Lanyun Zhu, Tianrun Chen, Deyi Ji, Jieping Ye, and Jun Liu. Llafs: When large language models meet few-shot segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3065–3075, 2024.
[106] Lanyun Zhu, Tianrun Chen, Qianxiong Xu, Xuanyi Liu, Deyi Ji, Haiyang Wu, De Wen Soh, and Jun Liu. Popen: Preference-based optimization and ensemble for lvlm-based reasoning segmentation. *arXiv preprint arXiv:2504.00640*, 2025.
[107] Lanyun Zhu, Tianrun Chen, Jianxiong Yin, Simon See, and Jun Liu. Continual semantic segmentation with automatic memory sample selection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3082–3092, 2023.
[108] Lanyun Zhu, Tianrun Chen, Jianxiong Yin, Simon See, and Jun Liu. Addressing background context bias in few-shot segmentation through iterative modulation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024.
[109] Lanyun Zhu, Deyi Ji, Shiping Zhu, Weihao Gan, Wei Wu, and Junjie Yan. Learning statistical texture for semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 12537–12546, 2021.
[110] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In *Int. Conf. Comput. Vis.*, pages 593–602, 2019.

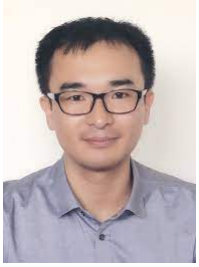
Lanyun Zhu received his Ph.D degree from the Information Systems Technology and Design (ISTD) pillar, Singapore University of Technology and Design in 2025, and B.E. degree from Beihang University, Beijing, China in 2020. His research interests are mainly focused on deep learning, computer vision, and multimodal learning. He is the reviewer of multiple top journals and conferences including TPAMI, TIP, TMM, TCSVT, IJCV, CVPR, ICCV, ECCV, ICML, NeurIPS and ICLR.

Tianrun Chen received the bachelor's degree in College of Information Science and Electronic Engineering, Zhejiang University and is pursuing the Ph.D degree with the College of Computer Science and Technology, Zhejiang University. He is the founder and the technical director of Moxin (Huzhou) Technology Co., LTD. His research interest includes computer vision and its enabling applications.

Jianxiong Yin currently is a Senior Deep Learning Solutions Architect with NVIDIA AI Technology Center. He was a researcher with Nanyang Technological University (NTU) from 2012 to 2016, during which he received ASEAN ICT Awards Gold Award, Datacenter Dynamics Award, ACM SIGCOMM 2013 travel grant and GTC 2015 presenter grant for his research work in cloud datacenter energy optimization, datacenter digital twin and distributed deep learning training system architecture.

Simon See received the Ph.D. degree in applied mathematics/engineering from the University of Salford. He is currently the Senior Director and the Chief Solutions Architect with NVIDIA Asia Pacific Professional Solution Group. He is also a Professor with Shanghai Jiaotong University, China, and Mahindra University, India. He is also the Chief Scientific Computing Advisor to BGI, China. His research interests are artificial intelligence, computer architecture and systems.

De Wen Soh received the B.S. degree in Mathematics from Stanford University. He received his Ph.D. degree in Electrical Engineering from Yale University under the supervision of Sekhar Tatikonda, where he worked on high-dimensional graphical model learning. He is currently an Assistant Professor in the Singapore University of Technology and Design. His research areas include machine learning, computer vision, natural language processing and statistical theory.



Jun Liu is a Professor and Chair in Digital Health at School of Computing and Communications in Lancaster University. He got the PhD degree from Nanyang Technological University in 2019. He is an Associate Editor of IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON BIOMETRICS, BEHAVIOR AND IDENTITY SCIENCE, *ACM Computing Surveys*, and *Pattern Recognition*. He has served as an Area Chair of CVPR, ECCV, ICML, NeurIPS, ICLR and MM.