Tackling Catastrophic Forgetting and Background Shift in Continual Semantic Segmentation

Arthur Douillard\(^1,2\), Yifu Chen\(^1\), Arnaud Dapogny\(^3\), and Matthieu Cord\(^1,4\)

Abstract—Deep learning approaches are nowadays ubiquitously used to tackle computer vision tasks such as semantic segmentation, requiring large datasets and substantial computational power. Continual learning for semantic segmentation (CSS) is an emerging trend that consists in updating an old model by sequentially adding new classes. However, continual learning methods are usually prone to catastrophic forgetting. This issue is further aggravated in CSS where, at each step, old classes from previous iterations are collapsed into the background. In this paper, we propose Local POD, a multi-scale pooling distillation scheme that preserves long- and short-range spatial relationships at feature level. Furthermore, we design an entropy-based pseudo-labelling of the background w.r.t. classes predicted by the old model to deal with background shift and avoid catastrophic forgetting of the old classes. Finally, we introduce a novel rehearsal method that is particularly suited for segmentation. Our approach, called PLOP, significantly outperforms state-of-the-art methods in existing CSS scenarios, as well as in newly proposed challenging benchmarks.

Index Terms—Continual Learning, Semantic Segmentation, Knowledge Distillation, Rehearsal Learning

1 INTRODUCTION

Computer vision went this last decade under a dramatic change from hand-crafted features \([1], [2]\) and machine learning models \([3]\) to deep convolutional neural networks (DCNNs) \([4]\). Among the existing tasks involving computer vision, semantic segmentation aims to assign a label to each pixel of an image. It allows the prediction of multiple objects in the same image, and moreover their exact position and shape. This task recently flourished \([5], [6], [7]\) with larger datasets with thousand of fully annotated images \([8], [9]\), increased computational power, and larger attention \([10]\). Unfortunately, the recent research in this area is often impracticable for real-life applications: they mostly need fully annotated data and require to be retrained from scratch if a new class is added to the dataset. Ideally, one would wish to regularly expand a dataset, only adding and labelling new classes and updating the model in accordance. This setup, referred here as Continual Semantic Segmentation (CSS), has emerged very recently for specialized applications \([11], [12], [13]\) before being proposed for general segmentation datasets \([14], [15], [16]\).

In particular, in this paper, we argue that two problems arise when performing CSS with DCNNs. The first one, inherited from continual learning, is called catastrophic forgetting \([17], [18], [19]\), and points to the fact that neural networks tend to completely and abruptly forget previously learned knowledge when learning new information \([20]\). Catastrophic forgetting presents a real challenge for continual learning applications based on deep learning methods. For some applications, storing previously seen data is not allowed for privacy reasons. This is a real challenge because the model can no longer be trained on the old categories with labeled data. The worse part is that the previously learned categories may not even exist in the future data, which requires the model to never forget these classes in the case of learning only once. In the case where storage of past data is allowed, we can use the past data to prevent forgetting. However, given the size of the dataset, fine-tuning over all past data is extremely costly in memory and time. Therefore, rehearsal methods are usually employed, which consist of storing some of the past examples and replaying them while learning new information. This problem is exacerbated in continual segmentation where the storage cost is high and images are partially labeled.

The second challenge, specific to CSS, is the semantic shift of the background class. In a traditional semantic segmentation setup, all object categories are predefined, and the “background” class contains pixels that do not belong to any of these classes. However, in CSS, the background contains pixels that do not belong to any of the current classes. Thus, for a specific learning step, the background can contain both future classes, not yet seen by the model, as well as old classes. Thus, if nothing is done to distinguish pixels belonging to the real background class from old class pixels, this background shift phenomenon risks exacerbating the catastrophic forgetting even further \([15]\). This issue also has an impact on the selection of the old data we want to store. Because some currently learned classes are annotated as background in the old data, this may degrade the performance of these classes if one naively treat them as background to fine-tune the current model.

We tackle the first challenge of catastrophic forgetting by designing a constraint enforcing a similar behavior between the old and current model. Specifically, we leverage intermediary representations of the convolutional networks to ensure that similar patterns are extracted through time. This feature-based constraints, called Local POD, fully exploits the global and local scale necessary to semantic segmentation through a multi-scale design. The second challenge, background shift, is greatly alleviated a confidence-based pseudo-labeling strategy to retrieve old class pixels within the background. For instance, if a current ground truth mask only distinguishes pixels from class sofa and
background, our approach allows to assign old classes to background pixels, e.g. classes person, dog or background (the semantic class). We name PLOP the model exploiting those two contributions. We then propose an extension called PLOPLong that aims to excel on long continual learning scenarios. This extension exploits cosine normalization to adapt the classifier and the Local POD resulting in an improving robustness to the discrepancy between old and new classes. Moreover, PLOPLong features a modified batch normalization which reduces the sensitivity of the model to moving statistics seen across tasks in continual learning. Finally, we investigate for the first time rehearsal learning for CSS. We propose an initial version based on rehearsing complete images which unfortunately is memory intensive. We then create the Object Rehearsal, a rehearsal method as performant but significantly more memory efficient.

From a practical point of view, our proposed methods (PLOP, PLOPLong, and Object Rehearsal) showed three important results. First, we achieve the state-of-the-art performance on several challenging datasets. Secondly, we propose several novel scenarios to further quantify the performances of CSS methods when it comes to long term learning, class presentation order and domain shift. Last but not least, we show that our model contributions largely outperform every CSS approach in these scenarios.

To sum it up, our contributions are four-folds:

- We propose a multi-scale spatial distillation loss to better retain knowledge through the continual learning steps, by preserving long- and short-range spatial statistics, avoiding catastrophic forgetting.
- We introduce a confidence-based pseudo-labeling strategy to identify old classes for the current background pixels and deal with background shift.
- We propose PLOPLong, a carefully designed refinement of our method for dealing with long CSS scenarios. The extension comes from an adaptation of PLOP’s classifier and Local POD distillation as well as batch renormalization for better handling of both catastrophic forgetting and background shift, respectively.
- We design a novel memory-efficient Object rehearsal learning procedure that consists in storing and carefully pasting objects through selective erasing of foreground objects.

Additionally, We show that PLOP significantly outperforms state-of-the-art approaches in existing scenarios and datasets for CSS, as well as in several newly proposed challenging benchmarks. Furthermore, we show that PLOPLong leads to superior performances on longer CSS scenarios. Last but not least, we show that performance of CSS models can be greatly improved where rehearsal learning is an option. In such cases, the proposed Object rehearsal allows to reach high accuracies with low memory footprint.

In this paper, we expand on our previous work published in conference venue [16]. The updated PLOP includes both model and experimental improvements. While keeping the losses introduced with PLOP, we incorporate PLOP with the batch renormalization technique [21] and a cosine classifier [22]. We structurally modify part of the network with these methods, which further mitigates forgetting on long stream of tasks compared to the original PLOP. We also investigate rehearsal learning for continual segmentation. Taking into account the background shift problem for replying and also to minimise the external memory needed to store past data, we propose a novel memory-efficient rehearsal method based on objects pasting. We extend our experiments with a class-incremental version of Cityscapes, which proved to be challenging, and we show how our model modifications handle better this dataset. Finally, we propose more ablations and compare rehearsal-based methods against a semi-supervised method using an external unlabeled dataset.

2 Related Work

Continual Semantic Segmentation is a relatively young field that started getting traction following [14], [15]. However, this field is at the intersection of many popular topics. Therefore, we start this section with an overview of recent advances in segmentation, continual learning, and a particular aspect of the latter: rehearsal learning. We then follow with a more in-depth discussion of existing approaches to CSS.

Semantic Segmentation methods based on Fully Convolutional Networks (FCN) [23], [24] have achieved impressive results on several segmentation benchmarks [8], [25], [26], [27]. These methods improve the segmentation accuracy by incorporating more spatial information or exploiting contextual information specifically. Atrous convolution [28], [29] and encoder-decoder architecture [30], [31], [32] are the most common methods for retaining spatial information. Examples of recent works exploiting contextual information include attention mechanisms [5], [6], [33], [34], [35], [36], [37], and fixed-scale aggregation [7], [28], [38], [39].

Continual Learning models suffer from catastrophic forgetting where old classes are forgotten, leading to a major drop of performance [17], [18], [19]. Multiple approaches exist to reduce this forgetting, among them some adapt the architecture continually to integrate new knowledge [40], [41], [42] or train together co-existing sub-networks [43] each specialized in one specific task [44], [45], [46]. It can be also possible to fix the knowledge drift in the classifier by recalibration [47], [48], [49], [50]. A popular approach consists in designing constraints/distillations that enforce the new model to exhibit a similar behavior as the previous model. These constraints can be defined directly on the weights [51], [52], [53], [54], the gradients [55], [56], the output probabilities [15], [57], [58], [59], or even intermediary representations [60], [61], [62], [63].

Rehearsal Learning [17] allows to replay a limited amount of samples from previous tasks in order to reduce forgetting. While popular for continual image classification [58], [59], [64], [65], [66], [67], [68], [69] or continual reinforcement learning [70], this approach has been fewly explored in continual segmentation. It can be explained in part because of the prohibitive memory cost of the large images used in segmentation (up to 1024 x 2048 vs 224 x 224). Another drawback comes the nature of continual segmentation where images at each step are partially labeled. This partial labelling will be different for each rehearsed image, leading to a damaging concept shift [71], [72]. Huang et al. [73] proposed after our PLOP a pseudo-rehearsal method for segmentation that while effective, incurs an important time overhead (up to 3x longer).

Continual Semantic Segmentation: Despite enormous progress in the two aforementioned areas respectively, segmentation algorithms are mostly used in an offline setting, while continual learning methods generally focus on image classification. Recent works extend existing continual learning methods [57], [60] for
specialized applications [11], [12], [13] and general semantic segmentation [14]. The latter considers that the previously learned categories are properly annotated in the images of the new dataset. This is an unrealistic assumption that fails to consider the background shift: pixels labeled as background at the current step are semantically ambiguous, in that they can contain pixels from old classes (including the real semantic background class, which is generally deciphered first) as well as pixels from future classes. Cermelli et al. [75] propose a novel classification and distillation losses. Both handle the background shift by summing respectively the old logits with the background logits and the new logits with the background. We argue that a distillation loss applied to the model output is not strong enough for catastrophic forgetting in CSS. Furthermore, their classification loss doesn’t preserve enough discriminative power w.r.t the old classes when learning new classes under background shift. We introduce our PLOP framework that solves more effectively those two aspects. Yu et al. [74] proposed to exploit an external unlabeled dataset in order to self-training through a pseudo-labeling loss; we show that our model while not designed with this assumption in mind PLOP framework that solves more effectively those two aspects. Moreover, background is often not considered [14] or only weakly tackled [15], while we propose to eliminate it through segmentation maps completion with pseudo-labeling. Finally, none of the work proposed rehearsal methods for CSS, while we propose a non-trivial based on image rehearsal and further improve it with a more data-efficient method based with object rehearsal.

3 Model
The model description is organized as follows: we first details the continual protocol and the notations. Then, we tackle the issue of catastrophic forgetting by designing an adapted distillation loss and we alleviate the background shift by proposing a uncertainty-based pseudo-labeling. Drawing ideas from the continual learning litterature to conceive a model for continual distillation losses. Both handle the background shift by summing respectively the old logits with the background logits and the new logits with the background. We argue that a distillation loss applied to the model output is not strong enough for catastrophic forgetting in CSS. Furthermore, their classification loss doesn’t preserve enough discriminative power w.r.t the old classes when learning new classes under background shift. We introduce our PLOP framework that solves more effectively those two aspects. Yu et al. [74] proposed to exploit an external unlabeled dataset in order to self-training through a pseudo-labeling loss; we show that our model while not designed with this assumption in mind PLOP framework that solves more effectively those two aspects. Moreover, background is often not considered [14] or only weakly tackled [15], while we propose to eliminate it through segmentation maps completion with pseudo-labeling. Finally, none of the work proposed rehearsal methods for CSS, while we propose a non-trivial based on image rehearsal and further improve it with a more data-efficient method based with object rehearsal.

3.1 Framework and notations
In Continual Semantic Segmentation (CSS), a model learns a dataset \( D \) in \( t = 1 \ldots T \) steps. At each step \( t \), the model has to learn a new set of classes \( C^t \), for which only a subset of data \( D_t \), that consists in a set of pairs \((T_t, S^t)\), is available. \( T_t \) denotes an RGB input image of size \( W \times H \times 3 \) and \( S^t \) the corresponding one-hot encoded ground-truth segmentation maps of size \( W \times H \times C^t \). If a pixel of \( T_t \) belongs to either one of the previous \( C^{t-1} \) or future classes \( C^{t+1:T} \), it is not labeled as such in \( S^t \) and shall instead be considered as belonging to the background class, denoted \( c_{bg} \). However, at test time, a model at step \( t \) must be able to discriminate between all the classes that have been seen so far, i.e. \( C_{1:t} \). This leads us to identify two major pitfalls in CSS: the first one, inherited from continual learning, is catastrophic forgetting [17], [18]. It suggests that a network will completely forget the old classes \( C_{1:t-1} \) when learning the new ones \( C^t \). Furthermore, catastrophic forgetting is aggravated by the second pitfall, specific to CSS, that we call background shift: at step \( t \), the pixels labeled as background are indeed ambiguous, as they may contain either old (including the real background class, predicted in \( C^1 \)) or future classes.

We define our model at step \( t \) as a composition of a feature extractor \( f_t(\cdot) \) (a ResNet 101 [77] backbone and a classifier \( g_t(\cdot) \). The output predicted segmentation map can then be written \( S^t = g_t \circ f_t(T) \) where \( f_t \) and \( g_t \) are \( \Theta^t \). The notations introduced in this paper are summarized in Table 1.

| Table 1: Notations used in this paper. |
|---------------------------------------------|
| Total number of tasks | \( T \) |
| Current task | \( t \) |
| Current classes | \( C^t \) |
| Previous classes | \( C^{1:t-1} \) |
| Future classes | \( C^{t+1:T} \) |
| Cardinality of a set of classes | \( |N_t| = \text{card}(C^t) - 1 \) |
| Image and Segmentation maps at task \( t \) | \( T^t \) and \( S^t \) |
| Feature extractor at task \( t \) | \( f_t(\cdot) \) |
| Classifier at task \( t \) | \( g_t(\cdot) \) |
| Learnable parameters at task \( t \) | \( \Theta^t \) |
| Predicted segmentation maps | \( S^t \) |
| Intermediary features at level \( l \) | \( f^t_l, l \in \{1, \ldots, L\} \) |

3.2 Overcoming catastrophic forgetting in CSS with local distillation
In this section, we propose to tackle the issue of catastrophic forgetting in continual learning in general and in CSS in particular. An effective method for doing so involves setting constraints between the old \( (g^{t-1} \circ f^{t-1}) \) and current \( (g^t \circ f^t) \) models. These constraints aim at enforcing a similar behavior between both models and in turn reduce the loss of performance on old classes. A common such constraint is based on applying knowledge distillation [57], [78] between the predicted probabilities of both models. When applied to CSS, such distillation loss must be carefully balanced to find a good trade-off between rigidity (i.e. too strong constraints, resulting in not being able to learn new classes) and plasticity (i.e. enforcing loose constraints, which can lead to catastrophic forgetting of the old classes).

In previous work [63] we designed POD, short for Pooled Output Distillation. Rather than solely constraining the output model probabilities, POD enforces consistency between intermediary statistics of both models. In practice, for a feature map \( x \), we define a POD embedding \( \Phi(\cdot) \) as:

\[
\Phi(x) = \left[ \frac{1}{W} \sum_{w=1}^{W} x[w, \cdot] \mid \frac{1}{H} \sum_{h=1}^{H} x[h, \cdot, \cdot] \right] \in \mathbb{R}^{(H+W) \times C},
\]

where \( [\cdot \mid \cdot] \) denotes concatenation over the channel axis. The POD embedding is thus computed as the concatenation of the \( H \times C \) width-pooled slices and the \( W \times C \) height-pooled slices of \( x \) and captures long-range statistics across the whole features...
maps. The POD distillation loss then consists in minimizing the \( \mathcal{L}_2 \) distance between POD embeddings computed at several layers \( l \in \{1, \ldots, L\} \), w.r.t the current model parameters \( \Theta^l \):

\[
\mathcal{L}_{\text{pod}}(\Theta^l) = \frac{1}{L} \sum_{l=1}^{L} \| \Phi(f^l(I)) - \Phi(f_{\text{pseudo}}^{l-1}(I)) \|^2 .
\]  

(2)

POD yields state-of-the-art results in continual learning for image classification, especially when large numbers of tasks are considered, a case where the aforementioned plasticity-rigidity trade-off becomes even more crucial. Another interest arises in the context of CSS: the long-range statistics computed across an entire axis (horizontal or vertical) which reminds recent work on attention for segmentation \[10\], \[79\], \[80\] which aim to enlarge the receptive field through global attention/statistics \[10\].

In the frame of classification, it is, to a certain extent, necessary to discard spatial information through global pooling. However, conversely, semantic segmentation requires preservation of both long-range and short-range statistics, making a distillation loss such as POD suboptimal for that purpose.

Following this reflexion, and inspired by the multi-scale literature \[81\], \[82\], we design a distillation loss, called Local POD that retain the long-range spatial statistics while also preserving the local information. The proposed Local POD consists in computing the width and height statistics at different scales \( \{1/2^s\}_{s=0,1,\ldots} \), as illustrated in Fig. 1 (a). At a given level \( l \) of the features extractor, \( s \) POD embeddings are computed per scale \( s \) and concatenated:

\[
\Psi^s(x) = [\Phi(x^s_{0,0})]\ldots[\Phi(x^s_{-1,-1})] \in \mathbb{R}^{s \times (H+W) \times C} .
\]  

(3)

where \( \forall i = 0 \ldots s-1, \forall j = 0 \ldots s-1, x^s_{i,j} = x[iH/s : (i+1)H/s, jW/s : (j+1)W/s, :] \) is a sub-region of the embedding tensor \( x \) of size \( W/s \times H/s \). We then concatenate (along the channel axis) the Local POD embeddings \( \Psi^s(x) \) of each scale \( s \) to form the final embedding:

\[
\Psi(x) = [\Psi^1(x)]\ldots[\Psi^S(x)] \in \mathbb{R}^{(1+\ldots+S) \times (H+W) \times C} .
\]  

(4)

Similarly to POD, we compute Local POD embeddings for every layer \( l \in \{1, \ldots, L\} \) of both the old and current models. The resulting loss is thus:

\[
\mathcal{L}_{\text{LocalPOD}}(\Theta^l) = \frac{1}{L} \sum_{l=1}^{L} \| \Psi(f^l(I)) - \Psi(f_{\text{pseudo}}^{l-1}(I)) \|^2 .
\]  

(5)

Thus, notice that the first scale of Local POD \((1/2^0)\) is similar to the original POD and models long-range dependencies across the entire image. The subsequent scales \( s = 1/2^1, 1/2^2, \ldots \), enforce short-range dependencies. Thus, the proposed Local POD tackles the problem of catastrophic forgetting by modeling and preserving long and short-range statistics between the old and current models, throughout the CSS steps.

### 3.3 Pseudo-labeling to fix background shift

In addition to catastrophic forgetting, a successful CSS approach shall handle the background shift problem, thus shall take into account the ambiguity of pixels labelled as background at each step. We propose a pseudo-labeling strategy that “completes” the ambiguous background labels. Pseudo-labeling \[83\] is commonly used in domain adaption for semantic segmentation \[84\], \[85\], \[86\], \[87\] where a model is trained to match both the labels of a source dataset and the pseudo-labels (usually obtained using the same predictive model, in a self-training fashion) of an unlabeled target dataset. In this case, the knowledge acquired on the source dataset helps the model to generate labels for the target dataset. In the frame of CSS, at each step, we use the predictions of the old model to decipher previously seen classes among the ambiguous background pixels, as illustrated in Fig. 1. The pseudo-labeling rely on the previous model which can be uncertain for some
pixels due to inherent bias to the optimization and because of the forgetting. Therefore in order to avoid propagate errors through incorrect pseudo-labels, we filter out the most uncertain ones based on a adaptive entropy-based threshold.

Formally, let \( N^t = \text{card}(C^t) - 1 \) the cardinality of the current classes excluding the background class. Let \( S^t \in \mathbb{R}^{W,H,N_1+N_2+\cdots+N_t} \) denotes the predictions of the current model (which includes the real background class, all the old classes as well as the current ones). We define \( S_{\tilde{t}}^t \in \mathbb{R}^{W,H,1+N_2+\cdots+N_t} \) the target as step \( t \), computed using the one-hot ground-truth segmentation map \( \tilde{S}^t \in \mathbb{N}^{W,H,1+N_t} \) at step \( t \) as well as pseudo-labels extracted using the old model predictions \( \hat{S}^{t-1} \in \mathbb{R}^{W,H,1+N_2+\cdots+N_t-1} \) as follows:

\[
\hat{S}_{\tilde{t}}^t(w, h, c) = \begin{cases} 1 \text{ if } S^{t}(w, h, c_{\text{bg}}) = 0 \text{ and } c = \arg\max_{c' \in C^t} S^{t}(w, h, c') \\ 1 \text{ if } S^{t}(w, h, c_{\text{bg}}) = 1 \text{ and } c = \arg\max_{c' \in C_{\tilde{t}}^{t-1}} S^{t-1}(w, h, c') \\ 0 \text{ otherwise.} 
\end{cases}
\]  

In other words, in the case of non-background pixels we simply copy the ground truth label. Otherwise, we use the class predicted by the old model \( g^{t-1}(f^{t-1}() \cdot .) \). This pseudo-label strategy allows to assign each pixel labelled as background his real semantic label if this pixel belongs to any of the old classes. However, pseudo-labeling all background pixels can be unproductive, e.g. on uncertain pixels where the old model is likely to fail. Therefore we only keep pseudo-labels where the old model is deemed “confident” enough. Eq. 6 is modified to take into account this uncertainty:

\[
\tilde{S}^t(w, h, c) = \begin{cases} 1 \text{ if } S^t(w, h, c_{\text{bg}}) = 0 \text{ and } c = \arg\max_{c' \in C^t} S^t(w, h, c') \\ 1 \text{ if } S^t(w, h, c_{\text{bg}}) = 1 \text{ and } c = \arg\max_{c' \in C_{\tilde{t}}^{t-1}} \tilde{S}^{t-1}(w, h, c') \text{ and } u < \tau_c \\ 0 \text{ otherwise.} 
\end{cases}
\]

By notation abuse, \( u \) is function \( u(S^t(w, h)) \) that measures the uncertainty of the current model given a pixel \( i \in I(w, h) \). \( \tau_c \) denotes a class-specific uncertainty threshold. Hence, in the case where the old model is uncertain \( (u \geq \tau_c) \) about some pixels, they will be ignored in the final classification loss. Our framework is agnostic to the type of uncertainty used, but in practice we define it as the entropy. Therefore we use for \( u \) the current model’s per-pixel entropy \( u(S^t(w, h)) = -\sum_{c \in C^t} S^t(w, h, c) \log S^t(w, h, c) \). Likewise, the class-specific threshold \( \tau_c \) is computed from the median entropy of the old model over all pixels of \( D^t \) predicted the class \( c \) for all \( c \in C_{\tilde{t}}^{t-1} \) as proposed in [87]. Consequently, the cross-entropy loss with pseudo-labeling of the old classes can be written as:

\[
\mathcal{L}_{\text{pseudo}}(\Theta^t) = -\frac{\nu}{WH} \sum_{w,h} \sum_{c \in C^t} \tilde{S}(w, h, c) \log \hat{S}^t(w, h, c). 
\]

We reduce the normalization factor \( WH \) by as much discarded pixels. To avoid giving disproportional importance to the pixels belonging to new classes (which aren’t discarded), we introduce in Eq. 8 an adaptive factor \( \nu \), which is the ratio of accepted old classes pixels over the total number of such pixels. i.e. if most if most of the image is uncertain, the overall importance of the image relative to other images in the batch is reduced. The overall behavior of our pseudo-labeling is illustrated in Fig. 1.

We call our final model PLOP (standing for Pseudo-labeling and LOcal Pod). PLOP’s final loss is a weighted combination of Eq. 5 and Eq. 8:

\[
\mathcal{L}(\Theta^t) = \mathcal{L}_{\text{pseudo}}(\Theta^t) + \lambda \mathcal{L}_{\text{localPod}}(\Theta^t), 
\]

with \( \lambda \) an hyperparameter. PLOP, while already very competitive, can face difficulties when dealing with long continual settings, i.e. for which the number of steps grows larger. For this reason, we propose PLOPLong, an extension of PLOP for dealing with such cases.

### 3.4 PLOPLong: a specialization for long settings

A first axis of improvement consists in correcting the forgetting located at the classifier level. Indeed, the classifier weights tend to be specialized for the last classes to the detriment of older classes [60]. Several means exist to correct this bias [22], [47], [48], [49] and we choose the cosine normalization. In practice, we replace the classifier by a cosine classifier [22], where the final inner product is discarded in favor of a cosine similarity. By doing so, all class weights—both old and new—have a constant magnitude of 1, which reduces drastically the bias towards new classes. The classifier \( g^t \) is in segmentation a pointwise \((1 \times 1) \) kernel convolution which doesn’t alter the spatial organisation but maps the \( ch \) features channels to \( C_{\tilde{t}} \) channels, one per class to predict. This pointwise convolution can be seen as a fully-connected layer that is apply independently to each pixel. Therefore, the classifier \( g^t \) has \( \{ \theta_c^t \in \mathbb{R}^{ch} | \forall c \in C_{\tilde{t}} \} \). The cosine normalization can then be express as:

\[
\hat{S}(w, h, c) = \frac{\alpha(\theta^t_c, h(w, h))}{||\theta^t_c||^2 ||h(w, h)||^2} 
\]

with \( \alpha \) a learned scalar parameter initialized to 1 and helping the convergence, and \( h \in \mathbb{R}^{W \times H \times ch} \) the final features embedding before the classifier. First used in continual learning by Hou et al. [60], this classifier has more recently be adopted by continual few-shot segmentation [75] or multi-modes continual learning [63]. New cosine classifier weights for new classifier can then be initialized with weight imprinting [88] as recently adapted for segmentation by Cermelli et al. [75].

As second improvement we propose to exploit the cosine normalization for intermediary features: the comparison between the Local POD embeddings of the previous model \( f^{t-1} \) with the current \( f^t \) can be too constrained. We relax the constraints by imposing not a low L2 distance between both embeddings, but rather a high cosine similarity. We simply need to alter Eq. 5 by replacing the \( \Phi(\cdot) \) operator by \( \tilde{\Phi}(\cdot) = \Phi(\cdot)/||\Phi(\cdot)||^2 \). However note that the features from level \( l \) given to level \( l + 1 \) are not normalized. The normalization only happens for the Local POD embeddings.

The third axis of improvement focuses the Batch Normalization [89], an almost ubiquitous component to deep computer vision models. It normalizes internal representation with the batch mean \( \mu_B \) and batch standard deviation \( \sigma_B \) (resp. running mean \( \mu \) and running std \( \sigma \)) during training (resp. testing). Formally for a all batch normalization layer taking an input \( x \) and producing an output \( y \):

\[
y = x - \frac{\mu_B}{\sigma_B} \cdot \gamma + \beta 
\]
with $\gamma$ and $\beta$ two learned parameters. The drawback of this normalization layer is its assumption that the data is sampled i.i.d. which is not the case in continual segmentation where multiple shifts [71], [72], [90], happen in the training tasks, different from the testing tasks. Lomonaco et al. [91] and Cermelli et al. [75] proposed to replace the batch normalization by the batch renormalization [21]:

$$y = \frac{x - \mu_B}{\sigma_B} \cdot r + d,$$

where $r = \frac{\sigma_B}{\sigma}$, $d = \frac{\mu_B - \mu}{\sigma}$

(12)

Intuitively, batch renormalization avoids the discrepancy between training and testing of the batch normalization. Furthermore, following [75], we freeze during training the statistics ($\mu$ and $\sigma$) after the first task to avoid harmful statistics drifts.

All three improvements further increase performance for long series of tasks: (1) Cosine classifier reduces the increased bias between recent and old classes. (2) A cosine-based Local POD relax the constraints in order to learn correctly the bigger number of new classes. (3) Frozen BatchReNorm reduces the inherent drift of statistics that grew larger as the number of tasks increase.

### 3.5 Rehearsing previous data for CSS

Neither the proposed PLOP or PLOPLong did make use of previously seen data $\{D_1, \ldots, D_{t-1}\}$ when considering step $t$. In this section, we explore how to further improve CSS performance if a model is now allowed to rehearse a limited amount of previous data. We first consider a traditional approach, namely Image Rehearsal, that we adapt to the CSS setting, and then propose a more efficient method, that we call Object Rehearsal.

#### 3.5.1 Image Rehearsal

We first consider rehearsing a limited amount of images from previous tasks during the current task. Contrary to image classification, in CSS, images can have multiple labels if several objects (e.g. car, sky) are present in the image. We therefore propose to select $M$ images for each class $c \in C^t$, resulting in $M \times |C^t|$. We ensure that the selected images are unique, thus because multiple classes can co-exist in the same image, the resulting amount of sampled classes may be above $M \times |C^t|$. We select the images with a simple random selection, which has been proved to generally be as good as more elaborate methods [59]. The memory footprint will grow until all tasks but the last are considered. Indeed, images stored from a previous task $t_o$ ($t_o < t$, with $t$ the current task) will only have the classes $C^{t_o}$ labeled, and the other classes folded into the background. Here again, pseudo-labeling (see Sec. 3.3) can complete the segmentation maps with both even older classes $C^{t_o-1}$ and newer classes $C^{t_o+1/2}$.

#### 3.5.2 Object Rehearsal

The main drawback of image rehearsal is that CSS images are usually large ($512 \times 512$ to $1024 \times 2048$) yet they are sparsely informative, as a significant part of the images consists in background pixels [92] (e.g. 63% of Pascal-VOC [25] pixels) or belongs to a majority class (e.g. 32% of Cityscapes [26] pixels are roads).

To address both problems, instead of storing whole images from the previous tasks $\{C_1, \ldots, t-1\}$, we propose to store an informative portion that we will mix with the images of the current task $t$. Image mixing is popular for classification [93], [94], [95], [96], [97], [98] yet, to the best of our knowledge, sees limited use for semantic segmentation [99], [100], [101], [102], [103], and has never been considered to design memory-efficient rehearsal learning systems. Formally, given an image $I$ and the corresponding ground truth segmentation maps $S^t$, we define a binary mask $\Pi_c$ such that $\forall c \in C^t$:

$$O_c = I \cap \Pi_c \text{ where } \Pi_c(x,y) = \begin{cases} 1 & \text{ if } S^t(x,y) = c, \\ 0 & \text{ otherwise} \end{cases} \quad (13)$$

Fig. 2: Our Object Rehearsal strategy. In task $t-1$, we select from $\{D_1, \ldots\}$ a limited amount of objects (here bus, bird, and dog), which will be then mixed in the images $\{I^t, \ldots\}$ from the current task $t$; after pasting, the other present objects are erased. Finally the current model $g^t \circ f^t$ will be given the concatenation of the the original images and the augmented images $\{I^t, \ldots\}$. 
with \( O_c \) the selected object for class \( c \). By nature, this patch is extremely sparse and can be efficiently stored on disk by modern compression algorithms [104]. The total memory footprint at task \( t \) is thus \( M \times |C^{t+1}| \) objects.

Then, when learning a task \( t > 1 \), the model will learn on both the task dataset \( \bigcup_{i=1}^{t} (I_i, S_i) \) and the object memory \( \bigcup_{c \in C^{t+1}} \bigcup_{i=1}^{M} O_{c,i} \). The latter will augment the former through object pasting. We augment each object by applying an affine transformation matrix [99]:

\[
T = \begin{bmatrix}
z \cos \alpha & z \sin \alpha & 0 \\
-z \sin \alpha & z \cos \alpha & 0 \\
0 & 0 & 1
\end{bmatrix}, \tag{14}
\]

with \( z \) a zoom factor, and \( \alpha \) an in-plane rotation angle. Note that we don’t translate the object as its original position is often a good prior: indeed, objects are usually located at the same location (e.g. pedestrian on the left and right sidewalk, car on the middle road, etc.) We abuse the notation by denoting \( T(.) \) the application of this transformation matrix. Once the data augmentation is done on the object (and its mask), we paste it on an image \( I^t \) that refers to the current task:

\[
I^t = I^t \odot \left(1 - T(\Pi)\right) + T(O_c) \odot T(\Pi_c). \tag{15}
\]

The pasting can result in local incoherence where the pixel of a cow is pasted on top or next to the pixel of a television. Naively the object borders can be smoothed into the image with a Gaussian filter. Unfortunately it results in imprecise contours which are important in segmentation [105].

This approach has already been envisioned as a form of data augmentation in semantic segmentation, although the gains were small and to avoid the noise induced by this pasting, the batch size is prohibitively large (up to 512) [103]. We propose to reduce interference between the pasted object and the destination images by selective erasing of the surrounding pixels. Indeed, given a binary matrix \( \Xi(\Pi, S) \) of the same dimension as \( I \) and \( \Pi \):

\[
I^{ext} = I^{t} \odot \left(1 - \Xi\right) + \kappa \odot \Xi. \tag{16}
\]

We replace the pixels erased according to the mask \( \Xi \) with a RGB color \( \kappa \in \mathbb{R}^3 \). This RGB vector could be chosen through in-painting [99] or be random noise, but in practice we choose a constant color gray. We also update accordingly the segmentation maps likewise:

\[
S^{ext} = S^{t} \odot \left(1 - \Xi\right) + 255 \odot \Xi, \tag{17}
\]

where 255 is the default value used in segmentation to ignore some pixel labels which won’t be counted in the classification loss Eq. 8. We illustrate our rehearsal strategy in Fig. 2.

### 4 Experiments

#### 4.1 Datasets, Protocols, and Baselines

To ensure fair comparisons with state-of-the-art approaches, we follow the experimental setup of [15] for datasets, protocol, metrics, and baseline implementations. Although, we also propose to evaluate on new datasets and on more challenging protocols. Furthermore, we explore in advanced experiments, for the first time, how rehearsal can improve performance in CSS.

**Datasets:** We evaluate our model on three datasets, summarized in Table 2: Pascal-VOC [25], Cityscapes [26] and ADE20k [8]. VOC contains 20 classes, 10,582 training images, and 1,449 testing images. Cityscapes contains 2975 and 500 images for train and test, respectively. Those images represent 19 classes and were taken from 21 different cities. ADE20k has 150 classes, 20,210 training images, and 2,000 testing images. All ablations and hyperparameters tuning were done on a validation subset of the training set made of 20% of the images. For all datasets, we use random resize and crop augmentation (scale from 80% to 110%), as well as random horizontal flip during training time. The final image size for Pascal-VOC and ADE20k is 512 × 512 while it is 512 × 1024 for Cityscapes.

**CSS protocols:** [15] introduced the Overlapped setting where only the current classes are labeled vs. a background class \( C^t \). Moreover, pixels can belong to any classes \( C^{t+1} \cup C^t \cup C^{t+1:T} \) (old, current, and future). Those two constraints make this setting both challenging and realistic, as in a real setting there isn’t any oracle method to exclude future classes from the background. In all our experiments, we respect the Overlapped setting. While the training images are only labeled for the current classes, the testing images are labeled for all seen classes. We evaluate several CSS protocols for each dataset, e.g. on VOC 19-1, 15-5, and 15-1 respectively consists in learning 19 then 1 class (\( T = 2 \) steps), 15 then 5 classes (2 steps), and 15 classes followed by five times 1 class (6 steps). The last setting is the most challenging due to its higher number of steps. Similarly, on Cityscapes 14-1 means 14 followed by five times 1 class (6 steps) and on ADE 100-50 means 100 followed by 50 classes (2 steps). We provide a summary of all these settings in Table 3.

**Metrics:** We compare the different models using traditional mean Intersection over Union (mIoU). Specifically, we compute mIoU after the last step \( T \) for the initial classes \( C^t \), for the incremented classes \( C^{2:T} \), and for all classes \( C^{1:T} \) (all). These metrics respectively reflect the robustness to catastrophic forgetting (the model rigidity), the capacity to learn new classes (plasticity), as well as its overall performance (trade-of between both). We also introduce a novel avg metric (short for average), which measures the average of mIoU scores measured step after step, integrating performance

| Dataset   | Setting     | Mode | # tasks | # base classes | # classes / inc. task |
|-----------|-------------|------|---------|----------------|-----------------------|
| Pascal-VOC| 19-1 class  | 2    | 19      | 1              |
|           | 15-5 class  | 2    | 15      | 5              |
|           | 15-1 class  | 6    | 15      | 1              |
|           | 10-1 class  | 11   | 10      | 1              |
| Cityscapes| 14-1 class  | 6    | 14      | 1              |
| ADE20k    | 100-50 class| 2    | 100     | 50             |
|           | 50-50 class | 2    | 50      | 50             |
|           | 100-10 class| 6    | 100     | 10             |
|           | 100-5 class | 11   | 100     | 5              |

**TABLE 2:** Description of the three datasets considered in this paper. For datasets without explicit background class, one is created based on unlabeled pixels.

| Dataset   | # classes | Background? | # train | # test | Image size |
|-----------|-----------|-------------|---------|--------|------------|
| Pascal-VOC| 20        | ✓           | 10k     | 1.5k   | 512 × 512  |
| Cityscapes| 19        | X           | 3k      | 0.5k   | 512 × 1024 |
| ADE20k    | 150       | X           | 20k     | 2.0k   | 512 × 512  |

**TABLE 3:** Description of the 12 different benchmarks evaluated in this paper. For some of these, each task brings new classes while, for others, it comes with new domains.
over the whole continual learning process. We stress that all four metrics are important and we shouldn’t disregard one for the other: a model suffering no forgetting but not able to learn anything new is useless.

**Baselines:** We benchmark our model against the latest state-of-the-arts CSS methods ILT [14], MiB [15], GIFS [75], and SDR [76]. Note that while GIFS was created by Cermelli et al. [75] for continual few-shots segmentation, we adapt it for the more general task of CSS. Unless stated otherwise, all the results are excerpted from the corresponding papers. We also evaluate general continual models based on weight constraints (PI [54], EWC [51], and RW [53]) and knowledge distillation (LwF [57] and LwF-MC [58]).

Moreover, unless explicitly stated otherwise, all the models (ours included), do not use rehearsal learning.

**Implementation Details:** As in [15], we use a Deeplab-V3 [106] architecture with a ResNet-101 [77] backbone pretrained on ImageNet [107] for all experiments and models but SDR [76] which used a Deeplab-V3+ [108]. For all datasets, we set a maximum threshold for the uncertainty measure of Eq. 7 to \( \tau = 1e - 3 \). We train our model for 30, 60, and 30 epochs per CSS step on Pascal-VOC, ADE20k, and Cityscapes, respectively, with an initial learning rate of 1e−2 for the first CSS step, and 1e−3 for all the following ones. Note that for Cityscapes, the first step is longer with 50 epochs. We reduce the learning rate exponentially with a decay rate of 9e−1. We use SGD optimizer with 9e−1 Nesterov momentum. The Local POD weighting hyperparameter \( \lambda \) is set to 1e−2 and 5e−4 for intermediate feature maps and logits, respectively. Moreover, we multiply this factor by the adaptive weighting \( \sqrt{C_f/C_i} \) introduced by [60] that increases the strength of the distillation the further we are into the continual process. For all feature maps, Local POD is applied before ReLU, with squared pixel values, as in [63], [109]. We use 3 scales for Local POD: 1, 1/2, and 1/4, as adding more scales experimentally brought diminishing returns. For PLOPLong only, we L2-normalize all POD embeddings before distilling them as also done on [63]. Furthermore, for PLOPLong, the gradient norm-clipping is set at 1.0 for Pascal-VOC and 2.0 for Cityscapes. We use a batch size of 24 distributed on two 12Go Titan Xp GPUs. Contrary to many continual models, we don’t have access to any task id in inference, therefore our setting/strategy has to predict a class among the set of all seen classes.

### 4.2 Quantitative Evaluation

#### 4.2.1 Pascal VOC 2012

Table 4 shows quantitative experiments on VOC 19-1, 15-5, and 15-1. both the proposed PLOP and PLOPLong outperform state-of-the-art approaches, MiB [15], SDR [76], and GIFS [75] on all evaluated settings by a significant margin. For instance, on 19-1, the forgetting of old classes (1-19) is reduced by 4.39 percentage points (p.p) while performance on new classes is greatly improved (+13.76 p.p), as compared to the best performing method so far, MiB [15]. On 15-5, our model is on par with our re-implementation of MiB, and surpasses the original paper scores [15] by 1 p.p. On the most challenging 15-1 setting, general continual models (EWC and LwF-MC) and ILT all have very low mIoU. While MiB shows significant improvements, PLOP still outperforms it by a wide margin. Furthermore, PLOP also significantly outperform the best performing approach on this setting, GIFS [75] (+6.11 p.p). Furthermore, mIoU for the joint model is 77.40%, thus PLOP narrows the gap between CSS and joint learning on every CSS scenario. The average mIoU is also improved (+5.78 p.p) compared to GIFS, indicating that each CSS step benefits from the improvements related to our method. This is echoed by Fig. 3, which shows that while mIoU for both ILT and MiB deteriorates after only a handful of steps, PLOP’s mIoU remains very high throughout, indicating improved resilience to catastrophic forgetting and background shift. Last but not least, while PLOP performs better than PLOPLong on short setups (e.g. 19-1, 15-5), PLOPLong performs better, however, on longer CSS

| Method          | 0-19 (2 tasks) | 15-5 (2 tasks) | 15-1 (6 tasks) |
|-----------------|----------------|----------------|----------------|
|                 | 0-15           | 16-20          | 19-1           |
|                 | 0-15           | 16-20          | 19-1           |
| Fine Tuning†    | 6.08           | 2.10           | 0.20           |
| PI† [54]        | 7.50           | 1.60           | 0.00           |
| EWC† [51]       | 26.90          | 24.30          | 0.30           |
| RW† [53]        | 23.30          | 16.60          | 0.00           |
| LwF† [57]       | 51.20          | 58.90          | 1.00           |
| LwF-MC† [58]    | 64.40          | 58.10          | 6.40           |
| ILT [14]        | 67.10          | 66.30          | 4.90           |
| ILT [14]        | 67.75          | 67.08          | 8.75           |
| MiB† [15]       | 70.20          | 75.50          | 35.10          |
| MiB [15]        | 71.43          | 76.37          | 34.22          |
| SDR‡ [76]       | 71.30          | 76.30          | 47.30          |
| GIFS [75]       | 57.88          | 23.61          | 59.36          |
| PLOP            | **75.35**      | **75.73**      | **72.00**      |
| PLOP Long       | **74.75**      | **75.35**      | **71.66**      |

Table 4: Continual Semantic Segmentation results on Pascal-VOC 2012 in Mean IoU (%). †: results excerpted from [15], ‡ from [76]. Other results comes from re-implementation.
TABLE 5: Continual Semantic Segmentation results on Cityscapes 14-1 in Mean IoU (%).

| Method  | 1-14 | 15-19 | all   | avg  |
|---------|------|-------|-------|------|
| MiB [15] | 55.11 | 12.91 | 44.56 | 49.76 |
| PLOP    | 56.59 | 13.07 | 45.71 | 51.28 |
| PLOPLong | 58.60 | 15.04 | 47.71 | 54.31 |

benchmarks such as 15-1 (+2.81 p.p.).

4.2.2 Cityscapes
We also validate our method on Cityscapes in Table 5. In order to design a setting similar to Pascal-VOC 15-1 (6 tasks), we design the 14-1 setting. Also, we simulate a background class by folding together the unlabeled classes. Here again, PLOP performs slightly better than MiB (+2.48 p.p on the old classes, +0.16 p.p on the new ones, +1.15 p.p on average). Moreover, PLOPLong performs significantly better (+3.59 p.p on the old classes, +2.13 p.p on the new ones, +3.15 p.p on average), indicating better robustness to both catastrophic forgetting and background shift, especially when considering long continual learning setups. To more precisely assess this phenomenon, we investigate the performance of both methods when dealing with longer task learning sequences.

4.2.3 ADE20k
Table 6 shows experiments on ADE 100-50, 100-10, and 50-50. This dataset is notoriously hard, as the joint model baseline mIoU is only 38.90%. ILT has poor performance in all three scenarios. PLOP shows comparable performance with MiB on the short setting 100-50 (only 2 tasks), improves by 1.09 p.p on the medium setting 50-50 (3 tasks), and significantly outperforms MiB with a wider margin of 2.35 p.p on the long setting 100-10 (6 tasks). In addition to being better on all settings, PLOP showcased an increased performance gain on longer CSS (e.g. 100-10) scenarios, due to increased robustness to catastrophic forgetting and background shift.

4.2.4 Longer Continual Learning
We argue that CSS experiments should push towards more steps [59], [63], [91], [110] to quantify the robustness of approaches w.r.t. catastrophic forgetting and background shift. We introduce two novel and much more challenging settings with 11 tasks, almost twice as many as the previous longest setting. We report results for VOC 10-1 in Table 7 (10 classes followed by 10 times 1 class) and ADE 100-5 in Table 8 (100 classes followed by 10 times 5 classes). The second previous State-of-the-Art method, ILT, has a very low mIoU (< 6 on VOC 10-1 and practically null on ADE 100-5). Furthermore, the gap between PLOP and MiB is even wider compared with previous benchmarks (e.g. ×3.6 mIoU on VOC for mIoU of base classes 1-10), which confirms the superiority of PLOP when dealing with long continual processes. Furthermore, in such a case, PLOPLong really shines, bringing significant improvements (+17.03 p.p on the old classes, +3.05 p.p on the new classes, +10.42 p.p on average) due to combination of cosine normalization (both on the classifier and Local POD) and its frozen BatchReNormalization.

4.3 Object Rehearsal
Pascal-VOC 15-1: We now allow CSS models to store information from the previous steps and classes: in such a case, the overall model efficiency can be understood as to what extent it allows to find good trade-off between its accuracy (as measured by the aforementioned standard metrics) and the memory footprint of the stored images or objects. Methods such as HRHF [73] can not easily be understood in these terms as they do not store data, strictly speaking, but rather use deep inversion [111] techniques to generate synthetic data, which comes at a high time requirement: thus, we exclude this method from our comparisons. We first consider the challenging Pascal-VOC 15-1 setting in Table 9. We use PLOP and PLOPLong as baselines with 0 memory overhead, as both models only use data from the current task. Moreover, we compare with [74], where an unlabeled external dataset such as COCO [112] is used through pseudo-labelling to improve performance on Pascal-VOC, as both datasets present significant overlap in terms of classes and domains. We reimplemented their method and also considered PLOP in this configuration. Using the external COCO provide PLOP an important gain of mIoU for both old classes (+7 p.p.) and new classes (+24 p.p.). Furthermore, PLOP with COCO is significantly more performant than Yu et al. model (+5.43 p.p.) despite the latter was designed explicitly to use such unlabeled external dataset. Notice also how PLOPLong, without any kind of rehearsal, remains equivalent to [74] in terms of mIoU on 0-15 (72.00% vs 71.40%), despite the latter using a large pool of data (+20Go). Perhaps counter-intuitively the gain is located on the new classes (26.66% vs 40.00%) as the rehearsal effect has a regularizing effect leading to less over-predicting of the recent classes. A drawback of using COCO is that the visual domain is not exactly the same as VOC, therefore we also considered using PLOP with a rehearsal of the unlabeled VOC. Without surprise, it results in a much better overall performance (+3.88 p.p. compared to using COCO). Another setup that we consider is the image rehearsal paradigm, where, at each step, we keep a number of (randomly selected) images along with their segmentation map. These segmentation maps are however incomplete, due to the nature of the CSS problem (see Sec. 3.5): hence, we refer to this setting as partial VOC. We consider two amounts of images to keep: 10 images per class (22 Mb) and 1 image per class (2.2 Mb). PLOPLong largely benefits from this rehearsal, most notably on new classes (16-20) with a gain up to 16.56 p.p.. Finally, we compare our novel Object Rehearsal denoted by “object VOC” where we store either one object per class (0.26 Mb) or 10 objects per class (2.6 Mb). As shown on Table 9, the proposed object rehearsal, in addition to being significantly more memory efficient than whole image rehearsal (8.5× less space used), is equivalent or better in terms of mIoU especially for new classes (2.56 p.p.). The ensemble of our experiments proves that rehearsal, when the partial or missing labeling is carefully handled, can provide important performance gain. Furthermore, our novel object rehearsal manages to strike the best trade-off with a minimal memory overhead without impacting its performance.

Cityscapes: We propose more experiments with Object Rehearsal in Table 10 where we apply our model on Cityscapes 14-1. For the rehearsal, we sample either 10 images or object per class. Cityscapes images are particularly large (even resized to 512 × 1024) and "empty" (most of it is road and sky). Consequently the memory overhead is extremely important compared object rehearsal (117 vs 0.8). We show that both our novel image and object rehearsal improve performance of the already competitive PLOPLong, and with object rehearsal providing the large gain (+9 p.p. in all).
illustrates the relevance of our entropy-based uncertainty estimate. A comparison between these two results removed), which may constitute a performance ceiling of our method. Table 11b compares different classification losses when combined with our pseudo-labeling loss. As such, Cross-Entropy (CE) variants perform poorly, especially on new classes. UNCE, introduced in [15], improves by merging the background with old classes, however, it still struggles to correctly model the new classes, whereas our pseudo-labeling propagates more finely information of the old classes, while learning to predict the new ones, dramatically enhancing the performance in both cases. This penultimate row represents our full PLOP strategy.

**4.4.2 Robustness to class ordering**

Continual learning methods may be prone to instability. It has already been shown in related contexts [113] that class ordering can have a large impact on performance. Unfortunately, in real-world settings, the optimal class order can never be known beforehand: thus, the performance of an ideal CSS method should be as invariant to class order as possible. In all experiments done so far, this class order has been kept constant, as defined in [15]. We report results in Fig. 4 as boxplots obtained by applying 20 random permutations of the class order on VOC 15-1. We report results in all cases, PLOP surpasses MiB in term of avg mIoU. Furthermore, the standard deviation (e.g. 10% vs 5% on all) is always significantly lower, showing the excellent stability of PLOP compared with existing approaches. Furthermore, PLOPLong, while having more variance on the new classes, performs better overall as compared to PLOP, not to mention MiB. This is due to the fact that PLOPLong, due to improved distillation loss (at prediction and Local POD levels) and batch re-normalization that allows to better retain information of the old classes, which again is more conspicuous on longer CSS scenarios.

**4.4.3 Rehearsal alternatives**

Table 12 draws a comparison between several rehearsal alternatives in CSS. We split methods according to three criterions: “type” denotes whether we store a whole image, an object, or a patch. For both object and patch, we only rehearse a small amount of the images: in object, only the pixel’s objects are used while for patch, we use the pixel’s object and the close surrounding pixels that contain background information. We insert the rehearsal process. POD, proposed in [63], improves the results on the old classes, but not on the new classes (16-20). In fact, due to too much plasticity, POD model likely overfits and predicts nothing but the new classes, hence a lower mIoU. Finally, Local POD leads to superior performance (+20 p.p) w.r.t. all metrics, due to its integration of both long and short-range dependencies. This final row represents our full PLOP strategy.

**Table 6: Continual Semantic Segmentation results on ADE20k in Mean IoU (%).**

| Method | 0-100 | 101-150 | all | avg |
|--------|-------|---------|-----|-----|
| ILT [14] | 18.29 | 14.40 | 17.00 | 29.42 |
| MiB [15] | 40.52 | 17.17 | 32.79 | 37.31 |
| PLOP | 41.87 | 14.89 | 32.94 | 37.39 |

| Method | 51-150 | all | avg |
|--------|---------|-----|-----|
| ILT [14] | 3.53 | 2.85 | 3.21 |
| MiB [15] | 45.57 | 21.01 | 36.80 |
| PLOP | 48.83 | 20.99 | 34.82 |

| Method | 100-100 | 101-150 | all | avg |
|--------|---------|---------|-----|-----|
| ILT [14] | 0.11 | 3.06 | 1.09 | 12.56 |
| MiB [15] | 38.21 | 11.12 | 29.24 | 35.12 |
| PLOP | 40.48 | 13.61 | 31.59 | 36.64 |

**Table 7: Mean IoU on Pascal-VOC 2012 10-1.**

| Method | 0-10 | 11-20 | all | avg |
|--------|------|-------|-----|-----|
| ILT [14] | 7.15 | 3.67 | 5.50 | 25.71 |
| MiB [15] | 12.25 | 13.09 | 12.65 | 42.67 |
| PLOP | 44.03 | 15.51 | 30.45 | 52.32 |
| PLOPLong | 61.06 | 18.56 | 40.83 | 58.62 |

**Table 8: Mean IoU on ADE20k 100-5.**

| Method | 0-100 | 101-150 | all | avg |
|--------|-------|---------|-----|-----|
| ILT [14] | 0.08 | 1.31 | 0.49 | 7.83 |
| MiB [15] | 36.01 | 5.66 | 25.96 | 32.69 |
| PLOP | 39.11 | 7.81 | 28.75 | 35.25 |

4.4 Model Introspection

In this section, we perform component by component ablation study of PLOP, but also aim at understanding the behavior of the model, as well as comparing different rehearsal alternatives. Lastly we propose qualitative visualizations of the models predictions.

**4.4.1 PLOP Ablations**

We compare several distillation and classification losses on VOC 15-1 to stress the importance of the components of PLOP and report results in Table 11. All comparisons are evaluated on a val set made with 20% of the train set, therefore results are slightly different from the main experiments.

**Distillation comparisons:** Table 11a compares different distillation losses when combined with our pseudo-labeling loss. As such, UNKD introduced in [15] performs better than the Knowledge Distillation (KD) of [78], but not at every step (as indicated by the avg. value), which indicates instability during the training process. POD, proposed in [63], improves the results on the old classes, but not on the new classes (16-20). In fact, due to too much plasticity, POD model likely overfits and predicts nothing but the new classes, hence a lower mIoU. Finally, Local POD leads to superior performance (+20 p.p) w.r.t. all metrics, due to its integration of both long and short-range dependencies. This final row represents our full PLOP strategy.

**Classification comparisons:** Table 11b compares different classification losses when combined with our Local POD distillation loss. Cross-Entropy (CE) variants perform poorly, especially on new classes. UNCE, introduced in [15], improves by merging the background with old classes, however, it still struggles to correctly model the new classes, whereas our pseudo-labeling propagates more finely information of the old classes, while learning to predict the new ones, dramatically enhancing the performance in both cases. This penultimate row represents our full PLOP strategy. Also notice that the performance for pseudo-labeling is very close to Pseudo-Oracle (where the incorrect pseudo-labels are removed), which may constitute a performance ceiling of our uncertainty measure. A comparison between these two results illustrates the relevance of our entropy-based uncertainty estimate.
TABLE 9: Comparison of rehearsal-based methods on Pascal-VOC 2012 15-1 overlap in Mean IoU (%). We only consider the time overhead spent after the first task whose computation overhead is similar for all methods.

| Method                  | Rehearsal | Memory (Mb) | Time (Hours) | 0-15 | 16-20 | all | avg |
|-------------------------|-----------|-------------|--------------|------|-------|-----|-----|
| PLOP                    | —         | 0           | 1.8          | 65.12 | 21.11 | 54.64 | 67.21 |
| PLOPLong                | —         | 0           | 1.8          | 72.00 | 26.66 | 61.20 | 70.02 |
| Yu et al. [74]          | Unlabeled COCO | 20,000 | 7.0          | 71.40 | 40.00 | 65.60 | 75.85 |
| PLOP                    | Unlabeled COCO | 2,000  | 1.4          | 72.57 | 45.08 | 66.03 | 71.85 |
| PLOPLong                | Partial VOC | 2.2    | 2.6          | 74.18 | 38.87 | 66.81 | 72.48 |
| PLOPLong                | Object VOC  | 0.26 | 2.7          | 73.32 | 42.86 | 66.07 | 72.21 |
| Joint model             | —         | —           | —            | 79.10 | 72.60 | 77.40 | —   |

TABLE 10: Continual Semantic Segmentation results on Cityscapes 14-1 overlap in Mean IoU (%).

| Method          | Rehearsal | Memory | 1-14 | 15-19 | all | avg |
|-----------------|-----------|--------|------|-------|-----|-----|
| PLOPLong        | —         | 0      | 58.60| 15.04 | 47.71 | 54.31 |
| PLOPLong        | Partial   | 117.0  | 58.93| 19.55 | 49.09 | 55.74 |
| PLOPLong        | Object    | 0.8    | 57.82| 23.13 | 49.15 | 54.80 |

TABLE 11: Comparison studies on Pascal-VOC 2012 15-1 on a validation subset of 20% of the training set.

(a) Pseudo loss (Eq. 8) with different distillation losses.

| Distillation loss | 0-15 | 16-20 | all | avg |
|-------------------|------|-------|-----|-----|
| Knowledge Distillation | 29.72 | 4.42 | 23.69 | 49.18 |
| UNKD               | 34.85 | 5.26 | 27.80 | 46.39 |
| POD                | 43.94 | 4.82 | 34.62 | 53.35 |
| Local POD (Eq. 5)  | 63.06 | 17.92 | 52.31 | 65.71 |

(b) Local POD loss (Eq. 5) with different classification losses.

| Classification loss | 0-15 | 16-20 | all | avg |
|---------------------|------|-------|-----|-----|
| CE only on new      | 12.95 | 2.54 | 10.47 | 47.02 |
| CE                  | 33.80 | 4.67 | 26.87 | 50.79 |
| UNCE                | 48.46 | 4.82 | 38.62 | 53.19 |
| Pseudo (Eq. 8)      | 63.06 | 17.92 | 52.31 | 65.71 |
| Pseudo-Oracle       | 63.69 | 23.35 | 54.09 | 66.05 |

Rehearsal alleviates pseudo-labeling’s limitation: All previous methods, PLOP and PLOPLong included, didn’t consider rehearsal-learning. While popular in image classification, it has fewly been explored for continual segmentation. An attentive reader may wonder if rehearsal learning can have benefit given the fact that our pseudo-labeling can uncover “hidden” old classes in the images. Even a perfect pseudo-labeling (considered in Table 11b) may fail in some particular data situation where the class distribution of a step is almost dirac. We show in Fig. 5 the distribution of pixels per class during the 5th step of Pascal-VOC 15-1.

The new class to learn, train, is abundant but also almost alone but the class person. In this case, no amount of pseudo-labeling are replaced by gray pixels. We considered more complicated approaches such as in-painting [99] or textures filling [114], but were slightly less effective than our simpler solution. Overall, through careful design, we manage to get the best performance using our novel object rehearsal which was also the most memory-efficient.

Rehearsal alleviates pseudo-labeling’s limitation: All previous methods, PLOP and PLOPLong included, didn’t consider rehearsal-learning. While popular in image classification, it has fewly been explored for continual segmentation. An attentive reader may wonder if rehearsal learning can have benefit given the fact that our pseudo-labeling can uncover “hidden” old classes in the images. Even a perfect pseudo-labeling (considered in Table 11b) may fail in some particular data situation where the class distribution of a step is almost dirac. We show in Fig. 5 the distribution of pixels per class during the 5th step of Pascal-VOC 15-1. The new class to learn, train, is abundant but also almost alone but the class person. In this case, no amount of pseudo-labeling...
can uncover previous classes like bicycle or cat. Our proposed rehearsal learning is complementary to our pseudo-labeling loss where the former can reduce the weakness for the latter in some cases.

4.4.4 Visualization

Fig. 6 shows the predictions at the 6th and final step of VOC 15-1 for four different models: MiB, PLOP, PLOPLong, and PLOPLong with Object Rehearsal. MiB struggles to even find the correct classes: in the first image the car is predicted as train, in the second the plane as train, and in the third no classes are detected. On the other hand, PLOP always manages to pick up the present classes, although sometimes with imperfection. PLOPLong further refines results, and the addition of Object Rehearsal produces a sharp and near perfect masks compared to the ground-truths (GTs). We showcased the harmful effect of background shift in Fig. 7 where the class person is learned at 1st step and the class train at 5th step. MiB completely forgets the former class and overpredict the latter. The background shift is efficiently mitigated with pseudo-labeling as showed by the third row of PLOP. Our efficient proposition of object rehearsal allows further refinement of the predicted masks resulting in sharper boundaries.

5 Conclusion

Continual Semantic Segmentation (CSS) is an emerging but challenging computer vision domain. In this paper, we highlighted two major issues in CSS: catastrophic forgetting and background shift. To deal with the former, we proposed Local POD, a multi-scale distillation scheme that preserves both long and short-range spatial statistics between pixels. This lead to an effective balance between rigidity and plasticity for CSS, which in turns alleviate catastrophic forgetting. We then tackled background shift with an efficient uncertainty-based pseudo-labeling loss. It completes the partially-labeled segmentation maps, allowing the network to efficiently retain previously learned knowledge. Afterwards, We showed that carefully designed structural changes to the model could improve performance on long CSS scenarios, namely a cosine normalization adaptation of the classifier and Local POD followed by a modified batch normalization. Finally, we proposed to introduce rehearsal learning to CSS, one based on partially-labeled whole image rehearsal, and the other –much more memory-efficient– consisting in object rehearsal. The latter further refining our already effective model performance and allowing real world application of CSS with a stringent memory constraint. We evaluated the proposed PLOPLong, with or without object rehearsal, on three datasets and over twelve different benchmarks. In each, we showed that our model performs significantly better than all existing baselines. Finally, we qualitatively validate our model through extensive ablations in order to better understand the performance gain.

Acknowledgments

This work was granted access to the HPC resources of IDRIS under the allocation AD011011706 made by GENCI. We thank
Fig. 7: Visualization of the predictions of MiB, PLOP, PLOPLong, and PLOPLong + Object Rehearsal across time in VOC 15-1 on a test set image. At steps 1-4 only class person has been seen. At step 5, the class train is introduced, causing dramatic background shift. While MiB overfits on the new class and forgets the old class, PLOP is able to predict both classes correctly. PLOPLong and PLOPLong + Object Rehearsal further refine the predicted masks, resulting in much sharper boundaries.

Fabio Cermelli for the insightful discussions and his help with the BatchRenormalization code. We also thank Estelle Thou and Thomas Robert for the invaluable discussions and support.

REFERENCES

[1] D. G. Lowe, “Object recognition from local scale-invariant features,” in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 1999. 1
[2] F. Perronnin and C. Dance, “Fisher kernels on visual vocabularies for image categorization,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007. 1
[3] C. Cortes and V. Vapnik, “Support-vector networks,” Machine Learning, 2015. 1
[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems (NIPS), 2012. 1
[5] A. Tao, K. Sapra, and B. Catanzano, “Hierarchical multi-scale attention for semantic segmentation,” in arXiv preprint library, 2020. 1
[6] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, Z. Zhang, H. Lin, Y. Sun, T. He, J. Muller, R. Manmatha, M. Li, and A. Smola, “Resnest: Split-attention networks,” in arXiv preprint library, 2020. 1
[7] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in Proceedings of the IEEE European Conference on Computer Vision (ECCV), 2018. 1
[8] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ade20k dataset,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 1
[9] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kontschieder, “The mapillary vistas dataset for semantic understanding of street scenes,” in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017. 1
[10] H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, and L.-C. Chen, “Axial-deeplab: Stand-alone axial-attention for panoptic segmentation,” in Proceedings of the IEEE European Conference on Computer Vision (ECCV), 2020. 1
[11] F. Ozdemir, P. Fuernstahl, and O. Goksel, “Learn the new, keep the old: Extending pretrained models with new anatomy and images,” in International Conference on Medical Image Computing and Computer-Assisted Intervention, 2018. 1
[12] F. Ozdemir and O. Goksel, “Extending pretrained segmentation networks with additional anatomical structures,” in International journal of computer assisted radiology and surgery, 2019. 1
[13] O. Tarsa, Y. Tarabalka, and P. Alliez, “Incremental learning for semantic segmentation of large-scale remote sensing data,” in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2019. 1
[14] U. Michieli and P. Zanuttigh, “Incremental learning techniques for semantic segmentation,” in Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshop, 2019. 1
[15] F. Cermelli, M. Mancini, S. Rota Bulò, E. Ricci, and B. Caputo, “Modeling the background for incremental learning in semantic segmentation,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020. 1
[16] A. Robins, “Catastrophic forgetting, rehearsal and pseudorehearsal,” in Connection Science, 1995. 1
[17] A. French, “Catastrophic forgetting in connectionist networks,” in Trends in cognitive sciences, 1999. 1
[18] R. Kemker, M. McClure, A. Abitino, T. L. Hayes, and C. Kanan, “Measuring catastrophic forgetting in neural networks,” in Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2018. 1
[19] S. Ioffe, “Batch renormalization: Towards reducing minibatch dependence in batch-normalized models,” in Advances in Neural Information Processing Systems (NeurIPS), 2017. 1
[20] C. Luo, J. Zhan, X. Xue, L. Wang, R. Ren, and Q. Yang, “Cosine normalization: Using cosine similarity instead of dot product in neural networks,” in International Conference on Artificial Neural Networks, 2018. 1
[21] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 1
[22] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” in Proceedings of the International Conference on Learning Representations (ICLR), 2014. 2
[23] M. Everingham, S. M. Ali Eslami, L. Van Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” in International Journal of Computer Vision (IJCV), 2015. 1
[24] H. Caesar, J. R. R. Uijlings, and V. Ferrari, “Coco-stuff: Thing and stuff classes in context,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018. 2
[25] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” in IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2016. 2
[26] S. Mehta, M. Rastegari, A. Caspi, L. G. Shapiro, and H. Hajiislahi, “Efficient spatial pyramid of dilated convolutions for semantic segmentation,” in Proceedings of the IEEE European Conference on Computer Vision (ECCV), 2018. 2
[27] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), 2015. 2
Yifu Chen received his Ph.D. in 2020 from the Computer Science Laboratory (LIP6) of Sorbonne University, under the supervision of Prof. Matthieu Cord. His thesis is on deep learning methods for semantic image segmentation. His research also focuses on the visualization and explanation of deep neural networks, as well as on continual learning.

Arnaud Dapogny is a computer vision researcher at Datakalab in Paris. He obtained the Engineering degree from the Supélec engineering School in 2011 and the Masters degree from Sorbonne University, Paris, in 2013 with high honors. He also obtained his PhD at Institute for Intelligent Systems and Robotics (ISIR) in 2016 and worked as a post-doctoral fellow at LIP6. His works concern deep learning for computer vision and its application to automatic facial behavior as well as gesture analysis.

Matthieu Cord is full professor at Sorbonne University. He is also part-time principal scientist at Valeo.ai. His research expertise includes computer vision, machine learning and artificial intelligence. He is the author of more 150 publications on image classification, segmentation, deep learning, and multimodal vision and language understanding. He is an honorary member of the Institut Universitaire de France and served from 2015 to 2018 as an AI expert at CNRS and ANR (National Research Agency).