Continual Semantic Segmentation via Repulsion-Attraction of Sparse and Disentangled Latent Representations

Umberto Michieli Pietro Zanuttigh
Department of Information Engineering, University of Padova
{umberto.michieli, zanuttigh}@dei.unipd.it

Abstract

Deep neural networks suffer from the major limitation of catastrophic forgetting old tasks when learning new ones. In this paper we focus on class incremental continual learning in semantic segmentation, where new categories are made available over time while previous training data is not retained. The proposed continual learning scheme shapes the latent space to reduce forgetting whilst improving the recognition of novel classes. Our framework is driven by three novel components which we also combine on top of existing techniques effortlessly. First, prototypes matching enforces latent space consistency on old classes, constraining the encoder to produce similar latent representation for previously seen classes in the subsequent steps. Second, features sparsification allows to make room in the latent space to accommodate novel classes. Finally, contrastive learning is employed to cluster features according to their semantics while tearing apart those of different classes. Extensive evaluation on the Pascal VOC2012 and ADE20K datasets demonstrates the effectiveness of our approach, significantly outperforming state-of-the-art methods.

1. Introduction

Semantic segmentation is a challenging computer vision problem with many real-world applications ranging from robot sensing, to autonomous driving, video surveillance, virtual reality, and many others. For most applications, continuously improving the set of classes to be distinguished is a fundamental requirement. Current state-of-the-art semantic segmentation approaches are typically based on auto-encoder structures and on fully convolutional models [38] that are trained in a single-shot requiring all the dataset to be available at once. Indeed, existing architectures are not designed to incrementally update their inner classification model to accommodate new categories. This issue is well-known for deep neural networks and it is called catastrophic forgetting [41, 18, 20], as deep architectures fail to update their parameters for learning new categories while preserving good performance on the old ones.

Continual learning has been widely studied in image classification [32, 36] and object detection [56, 34], while has been tackled only recently in the semantic segmentation field [42, 58, 4, 33]. In this paper, we investigate class-incremental continual learning in semantic segmentation. Differently from the majority of previous approaches both in image classification [36, 51, 3] and semantic segmentation [58, 42, 4, 33], we do not mainly or solely rely on output-level knowledge distillation. In this work, we focus on latent space organization which has been only marginally investigated in the current literature, and we empirically prove it to be complementary to other existing techniques. The main idea is depicted in Fig. 1, where some of the latent space constraints are introduced. First, a prototype matching is devised to enforce features extraction consistency on old classes between the cumulative prototype computed using all previous samples and the current prototype (i.e., the prototype computed on the current batch only). In other
words, we force the encoder to produce similar latent representations for previously seen classes in the new steps. Second, a features sparsification constraint makes room in the latent space to accommodate novel classes. To further regularize the latent space, we introduce an attraction-repulsion rule similar in spirit to the recent advancements in contrastive learning. Finally, to enforce the decoder to preserve discriminability on previous categories during classification, we employ a targeted output-level distillation.

Although continual semantic segmentation has only been faced recently, it already comes with different experimental protocols depending on how the incremental data are considered (see Section 3.1): namely, sequential (new images are labeled with both new and old classes), disjoint (new images are labeled with only new classes, old classes are assigned to the background) and overlapped (new images are labeled with only new classes, images are repeated across training steps with different semantic maps associated to them). In this paper we devise a common framework which allows to tackle all these scenarios and can be applied in combination with previous techniques, which has never been attempted before. We evaluate on standard semantic segmentation datasets, like Pascal VOC2012 [16] and ADE20K [76], in many scenarios.

Summing up, the main contributions of this work are: 1) We investigate class-incremental learning in semantic segmentation, providing a common framework for different experimental protocols. 2) We explore the latent space organization and we propose complementary techniques with respect to the existing ones. 3) We propose novel knowledge preservation techniques based on prototypes matching, contrastive learning and features sparsity. 4) We benchmark our approach on standard semantic segmentation datasets outperforming state-of-the-art continual learning methods.

2. Related Work

**Continual Learning.** Deep learning models are prone to catastrophic forgetting [20, 30, 48], i.e., training a model with new information interferes with previously learned knowledge and typically greatly degrades performance. This phenomenon has been widely studied in image classification task and most of the current techniques fall into the following categories [10, 48]: regularization approaches [5, 32, 73, 13, 36], dynamic architectures [69, 64, 35], parameter isolation [17, 53, 40] and replay-based methods [66, 46, 55, 26]. Regularization-based approaches are far from the most widely employed and mainly come in two flavours, i.e., penalty computing and knowledge distillation [25]. Penalty computing approaches [73, 32, 32] protect important weights inside the models to prevent forgetting. Knowledge distillation [52, 66, 36, 13] relies on a teacher (old) model transferring or remembering knowledge related to previous tasks to a student model which is trained to learn also additional tasks. Parameter isolation approaches [40, 39] reserve a subset of weights for a specific task to avoid degradation. Dynamic architectures [64, 35] grow new branches for new tasks. Replay-based models exploit stored [3, 26, 51] or generated [66, 46, 55] examples during the learning process of new tasks.

**Continual Semantic Segmentation.** Nowadays, deep learning architectures have achieved outstanding results in semantic segmentation [19, 21]. Current approaches are based on fully convolutional models [38, 7, 6, 75, 72] and exploit various techniques to cope with multi-scale and spatial dependency. All these approaches, however, require training data and segmentation maps to be available at once (i.e., joint setting) and they experience catastrophic forgetting if new tasks (e.g., new classes to learn) are made available sequentially [42]. Hence, it emerged the need for continual approaches specifically targeted to solve the semantic segmentation task [47, 58, 42, 43, 33, 4]. Earlier works focus on the continual semantic segmentation problem in specific scenarios, e.g., in medical imaging [47] or remote sensing [58], extending standard image-level classification methods. More recently, standard semantic segmentation datasets and targeted methods have been proposed. In [42, 43] an exploration on knowledge distillation techniques is proposed to alleviate forgetting: the authors designed output-level and features-level distillation losses coupled with freezing the encoder’s weights. Klingner et al. [33] extend previous work not requiring old labels during the incremental steps and proposing class importance weighting to emphasize gradients on difficult classes. Cermelli et al. [4] study the distribution shift of the background class when it incorporates previous and/or future classes (disjoint and overlapped protocols, respectively). Background shift is addressed via unbiased versions of cross entropy and output-level knowledge distillation losses together with an unbiased weight initialization rule for the classifier. Nevertheless, previous works neglect accurate investigation of the latent space in continual learning.

**Latent Space Organization.** The analysis of the latent space organization is becoming crucial towards understanding and improvement of classification models [68, 49]. Recently, some attention has been devoted to latent regularization in continual image classification [1, 2, 27]. Besides this, one of the emerging paradigms is constrastive learning applied to visual representations. Dating back to [22], these approaches learn representations by contrasting positive against negative pairs and have been recently re-discovered for deep learning. Many works use a memory bank to store the instance class representation vector [67, 77, 59, 23, 44, 9], while some others explore the usage of in-batch negative samples instead [14, 71, 28, 31]. The contrastive learning objective proposed in this work moves from opposition of positive and negative pairs and
also recalls features clustering (if features belong to the same class) and separation (if features belong to different classes), which has been recently applied to adapt semantic segmentation models across domains [29, 37, 61].

Prototypes-based regularizing terms gained a great interest and, in particular, have been largely used in the literature of few-shot learning [15, 63, 60], to learn prototypical representations of each category, and domain adaptation, to enforce orthogonality [50, 65] or centroid matching [70, 12]. Finally, to minimize the interference among features we drive them to be channel-wise sparse. Only limited attention has been given on sparsity for deep learning architectures [2]; however, some prior techniques exist for domain adaptation on linear models exploiting sparse codes on a shared dictionary between the domains [54, 74].

Our work is the first combining together contrastive learning, sparsity and prototypes matching to regularize latent space for segmenting new categories over time.

### 3. Problem Definition and Setups

Before presenting the proposed strategies, we first introduce the semantic segmentation task, which assigns a class to each pixel in an image. We denote the input image space with \( \mathcal{X} \in \mathbb{R}^{H \times W \times 3} \) with spatial dimensions \( H \) and \( W \), the set of classes (or categories) with \( \mathcal{C} = \{c_i\}_{i=0}^{C-1} \) and the output space with \( \mathcal{Y} \in \mathbb{C}^{H \times W} \) (i.e., the segmentation map). Given a training set \( \mathcal{T} = \{(x_n, y_n)\}_{n=1}^{N} \), where \((x_n, y_n) \in \mathcal{X} \times \mathcal{Y} \), we aim at finding a map \( M \) from the input space to a pixel-wise class probability vector \( \mathcal{F} : \mathcal{X} \rightarrow \mathbb{R}^{H \times W \times C} \). Then, the output segmentation mask is computed as \( y_n = \arg \max_{c \in \mathcal{C}} M(x_n)[h, w, c] \), where \( h = 1, \ldots, H, w = 1, \ldots, W \) and \( M(x_n)[h, w, c] \) is the probability for class \( c \) in pixel \((h, w)\). Nowadays, \( M \) is typically some auto-encoder model made by an encoder \( E \) and a decoder \( D \) (i.e., \( M = E \circ D \)). We call \( \mathcal{F}_n = E(x_n) \) the feature map of \( x_n \), and \( y_n^* \) the downsampled segmentation map matching the spatial dimensions of \( \mathcal{F}_n \).

In the standard supervised setting it is assumed that the training set \( \mathcal{T} \) is available at once and the model is learned in one shot. In the continual learning scenario, instead, training is achieved over multiple iterations each carrying a novel category to learn and a subset of the training data. More formally, at each learning step \( k \) the previous label set \( \mathcal{C}_{k-1} \) is expanded with a set of novel classes \( \mathcal{S}_k \) forming a new label set \( \mathcal{C}_k = \mathcal{C}_{k-1} \cup \mathcal{S}_k \). Additionally, a new training subset \( \mathcal{T}_k \subset \mathcal{X} \times \mathcal{C}_k \) is made available and used to update the previous model into a new model \( \mathcal{M}_k \). Step \( k = 0 \) consists of a standard supervised training performed with only a subset of training data and classes. As in the standard incremental class learning scenario, we assume the different sets of new classes to be disjoint with the exception of the peculiar background class \( c_0 \), i.e., \( S_i \cap S_j = \{c_0\} \).

### 3.1. Experimental Protocols

Despite being quite a recent field, continual learning in semantic segmentation already comes in different flavors.

**Sequential:** this setup has been proposed in [42, 43]. Each learning step contains a unique set of images, whose pixels belong to classes seen either in the current or in the previous learning steps. At each step, labels for pixels of both old and novel classes are present.

**Disjoint:** this setup has been proposed in [4]. At each learning step, the unique set of images is identical to the sequential setup. The difference with respect to the sequential setup lies in the set of labels. At each step, only labels for pixels of novel classes are present, while the old ones are labeled as background in the ground truth.

**Overlapped:** this setup moves from the work of [56] for object detection and has been adapted to semantic segmentation in [4]. Each training step contains all the images that have at least one pixel of a novel class, with only the novel classes annotated while the rest is set to background. Differently from the other settings, here images may contain pixels of classes that will be learned in future learning steps, but they are labeled as background in the current step.

### 4. Method

In this section, we provide a detailed description of the core modules of the proposed method. Our approach leverages a contrastive learning objective applied over the feature representations, with novel prototypes matching and sparsity constraints. Specifically, features repulsion and attraction based on the semantic classes are enforced by grouping together features of the same class, while simultaneously pushing away those of different categories. We further regularize the distribution of latent representations by the joint application of prototypes matching and sparsity. While prototypes matching seeks for an invariant representation of the features extracted for the old classes, the sparsity objective encourages a lower volume of active feature channels from latent representations (i.e., it concentrates the energy of features on few dimensions) to free up space for new classes.

An overall scheme of our approach is shown in Fig. 2: the training objective is given by the combination of a cross-entropy loss (\( \mathcal{L}_{ce} \)) with the proposed modules. \( \mathcal{L}_{ce} \) is the usual cross-entropy loss for all the classes except for the background. The ground truth of the background, indeed, is not directly compared with its probabilities, but with the probability of having either an old class or the background in the current model [4]. Formally, at step \( k \) the background probabilities \( M(x_n)[h, w, c_0] \) are replaced by \( \sum_{c \in \mathcal{C}_{k-1}} M(x_n)[h, w, c] \). The rationale behind this is that the background class could incorporate statistics of previous classes in both the disjoint and overlapped protocols.

The other components are a prototypes matching target
(\mathcal{L}_{pm})$, a contrastive learning objective ($\mathcal{L}_{cl}$) and a sparsity constraint ($\mathcal{L}_{sp}$), which will be detailed in the following sections. The training objective is then computed as:

$$\mathcal{L}_{tot} = \mathcal{L}_{ce} + \lambda_{pm} \cdot \mathcal{L}_{pm} + \lambda_{cl} \cdot \mathcal{L}_{cl} + \lambda_{sp} \cdot \mathcal{L}_{sp}$$ (1)

where the $\lambda$ parameters balance the multiple losses and have been tuned using a validation set (see Section 5). Our aim is to seek for disentangled latent representations characterized by semantic-driven regularization and to show that this approach can achieve comparable or superior results with respect to standard regularization methods (e.g., output-level knowledge distillation). We further integrate the proposed framework with an output-level knowledge distillation objective [43] and we show that its effect is highly not overlapping, achieving increased accuracy. The training objective comprising an unbiased output-level distillation module is defined as:

$$\mathcal{L}_{tot} = \mathcal{L}_{tot}^\prime + \lambda_{kd} \cdot \mathcal{L}_{kd}$$ (2)

### 4.1. Prototypes Matching

Prototypes (i.e., class-centroids) are vectors that are representative of each category that appears in the dataset. During training, the features extracted by the encoder contribute in forming the latent prototypical representation of each class. To preserve the geometrical structure of the features of old classes we apply prototypes matching. Current prototypes $\hat{p}_c$ (i.e., computed on the current batch of images) are forced to be placed close to their representation learned from the previous steps $p_c$. We use the Frobenius norm $\| \cdot \|_F$ as metric distance [57, 45, 63]. More formally:

$$\mathcal{L}_{pm} = \frac{1}{|C_{k-1}|} \| p_c - \hat{p}_c \|_F \quad c \in C_{k-1}$$ (3)

The prototypes are computed in-place with a running average updated at each training step with supervision. At training step $t$ with batch $B$ of $B$ images, the prototypes are updated for a generic class $c$ as:

$$\hat{p}_c[t] = \frac{1}{B} \left[ (t-1) \hat{p}_c[t-1] + \sum_{n \in B} f_n \mathbb{1}[y_n^t = c] \right]$$ (4)

initialized to $\hat{p}_c[0] = 0 \forall c$. $f_n \in F_n$ is a generic feature vector and $y_n^t$ the corresponding pixel in $y_n^t$, $\mathbb{1}[y_n^t = c]$ indicates the pixels in $y_n^t$ associated to $c$ and $| \cdot |$ denotes cardinality.

We update the prototypes only when we have ground truth labels for that class to avoid incorporating the mutable statistics of the background class: we exclude the background from the incremental steps in the disjoint protocol (as it could contain old classes) and in the overlapped scenario (as it could contain old and future classes).

For the current batch $B$ of an incremental training stage, the current (or in-batch) prototypes $\hat{p}_c[t]$ are computed as:

$$\hat{p}_c[t] = \frac{1}{B} \sum_{x_n \in B} \begin{cases} \frac{\sum_{f \in F_n} f \mathbb{1}[y_n^t = c]}{| \mathbb{1}[y_n^t = c] |} & \text{if sequential} \\ \frac{\sum_{f \in F_n} f \mathbb{1}[y_n^t = c]}{|F_n|} & \text{otherwise} \end{cases}$$ (5)

where $\hat{z}_n^t$ (with pixels $\hat{z}_n^t$) is a pseudo-labeled segmentation map computed from the ground truth data by replacing the background region with the prediction from the previous model, since in the disjoint and overlapped protocols old classes are labeled as background. The difference between (4) and (5) lies in the usage of pseudo-labels: we use them in (5) to compute prototypes for old classes in the current batch since we may not have any label for them, but we avoid to use them in (4), since there is no need to update prototypes computed using the ground truth at previous steps with data from less reliable pseudo-labels.
4.2. Contrastive Learning

The second component is similar to recent contrastive learning [9, 59] and clustering [29, 37] approaches to constrain the latent space organization. The underlying idea is to structure the latent space in order to have features of the same category clustered near their prototype and at the same time to force prototypes to be far one from the other. We argue that this organization helps also in continual learning to mitigate forgetting and to facilitate the addition of novel classes, as features are clustered and there is more separation between the clusters. In formal terms, the constraint is defined by a loss \( \mathcal{L}_{cl} \) made of an attractive term \( \mathcal{L}_{cl}^a \) and a repulsive term \( \mathcal{L}_{cl}^r \), as follows:

\[
\mathcal{L}_{cl}^a = \frac{1}{|\mathcal{C}_j \in \mathcal{Y}_n|} \sum_{\mathcal{C}_j \in \mathcal{Y}_n} \sum_{k \in \mathcal{F}_n} \| (\mathbf{f}_i - \mathbf{p}_k) \|_F \quad (6)
\]

\[
\mathcal{L}_{cl}^r = \frac{1}{|\mathcal{C}_j \in \mathcal{Y}_n|} \sum_{\mathcal{C}_j \in \mathcal{Y}_n} \sum_{\mathcal{C}_k \in \mathcal{Y}_n, \mathcal{C}_k \neq \mathcal{C}_j} \| \mathbf{p}_{\mathcal{C}_j} - \mathbf{p}_{\mathcal{C}_k} \|_F \quad (7)
\]

The objective is composed of two terms: \( \mathcal{L}_{cl}^a \) measures how close features are from their respective centroids and \( \mathcal{L}_{cl}^r \) how spaced out prototypes corresponding to different semantic classes are. Hence, the effect provided by the loss minimization is twofold: firstly, feature vectors from the same class are tightened around class feature centroids; secondly, features from separate classes are subject to a repulsive force applied to feature centroids, moving them apart.

4.3. Features Sparsity

To enforce the regularizing effect brought by contrastive learning, we introduce a further feature-wise objective on the latent space. We propose a sparsity loss to decrease the number of active feature channels of latent vectors. First, to the latent space, we propose a sparsity loss to decrease the number of active feature channels of latent vectors. The underlying idea is to enforce the regularizing effect brought by contrastive learning, we introduce a further feature-wise objective on the latent space achieving simultaneously an invariant features extraction with respect to previous steps and an easier addition of novel categories.

\[
\mathcal{L}_{sp} = \frac{1}{|\mathcal{F}_n|} \sum_{\mathbf{f}_i \in \mathcal{F}_n} \sum_{j} \exp \left( \frac{\bar{\mathbf{f}}_{i,j}}{\mathbf{f}_i, \mathbf{g}_j \in \mathcal{F}_n} \right) \quad (9)
\]

We design the sparsity constraint as the ratio between a stretching function (we used the sum of exponentials) and a linear function (i.e., the sum) applied over each feature vector, which is minimized when the energy is concentrated in a few channels (since the normalized features assume values \( \leq 1 \)). The sparsity constraint is thus defined as:

While the contrastive learning objective forces features to lie within tight semantically-consistent well-distanced clusters, the sparsity constraint aims at narrowing the set of active channels with the aim of letting room for the representation of upcoming classes. In other words, by constraining features of the same classes to be tightly clustered and to be spaced apart from features of other classes and sparse, we can preserve geometrical space (few active channels) and expressiveness (division in well-separated clusters) for the latent representation of future classes. Empirically, we found entropy-based minimization methods in the latent space [62] to be less reliable for our task. In the Supplementary Material we show how to handle degenerate cases of (9) and an ablation on other sparsifying strategies.

4.4. Output-Level Knowledge Distillation

The last component of our work is an output-level knowledge distillation which we show to be complementary to the previously introduced strategies. Indeed, we add knowledge distillation on top of all the other components to transfer knowledge from the old model’s classifier to the current one. While previous constraints regularize the latent space achieving simultaneously an invariant features extraction with respect to previous steps and an easier addition of novel categories, output-level knowledge distillation directly acts on the classifier, to preserve its discriminative ability regarding old classes. In particular, we start from the preliminary considerations of [42, 43] and we employ the unbiased distillation proposed in [4] as natural extension to the case in which the background may contain other categories. In this case we avoid to re-normalize the probabilities from the previous step and, instead, we compare the background probability from the previous step with the probability of having either a new class or the background (this accounts for the fact that the background in the previous steps may include samples of the new classes, see [4]).

5. Training Procedure

To train and benchmark our approach we resort to two publicly available datasets following [56, 42, 43, 4]. The Pascal VOC 2012 [16] contains 10582 images in the training split and 1449 in the validation split (that we used for testing, as done by all competing works being the test set not publicly available). Each pixel of each image is assigned to one semantic label chosen among 21 different classes (20 plus the background). The ADE20K [76] is a large-scale dataset of 22210 images, 2000 of which form the validation split. The typical benchmark defined in [76] includes 150 classes, representing both stuff (e.g., sky, building) and object classes (e.g., bottle, chair), differently from VOC 2012.

The proposed strategy is agnostic to the backbone architecture. For the experimental evaluation of all the compared methods we use a standard Deeplab-v3+ [8] architecture with ResNet-101 [24] as backbone (differently from [2] for wider reproducibility) with output stride of 16. The back-
bone has been initialized using a pre-trained model on ImageNet [11] (see the Supplemental Material for a detailed discussion of the impact of different pre-training strategies). We optimize the network weights following [7] with SGD and with same learning rate policy, momentum and weight decay. The first learning step involves an initial learning rate of $10^{-2}$, which is decreased to $10^{-3}$ for the following steps as done in [56, 4]. The learning rate is decreased with a polynomial decay rule with power 0.9. In each learning step we train the models with a batch size of 8 for 30 epochs for Pascal VOC 2012 and a batch size of 4 for 60 epochs for ADE20K. Following [7], we crop the images to $512 \times 512$ during both training and validation and we apply the same data augmentation (i.e., random scaling the input images of a factor from 0.5 to 2.0 and random left-right flipping during training). In order to set the hyper-parameters of each method, we follow the same continual learning protocol of [10, 4], i.e., we used 20% of the training set as validation and we report the results on the original validation set of the datasets. We use Pytorch to develop and train all the models on a NVIDIA 2080 Ti GPU. The code is available at: https://lttm.dei.unipd.it/paper_data/SDR/.

6. Experimental Results

We evaluate the performance of our method (denoted in the tables with SDR, i.e., Sparse and Disentangled Representations) against some state-of-the-art continual learning frameworks. We report as a lower limit the performance of the naïve fine-tuning approach (FT), which consists in training the model on the newly available training data with no additional provisions, while the upper limit is given by the offline single-shot training (offline) on the whole dataset $\mathcal{T}$ and on all the classes at once. Then, we compare with 3 recent continual semantic segmentation schemes: ILT [42], which combines latent and output level knowledge distillation, CIL [33], which adds class importance weighting to output-level knowledge distillation, and MiB [4], which deals with the background distribution shift and proposes an unbiased weight initialization rule. We also report the results on LwF [36] (together with its single-headed version LwF-MC [51]), that according to [4] is the best performing continual image classification algorithm when adapted to semantic segmentation. For a fair comparison, all the methods have been re-trained with a standard Deeplab-v3+ [8] architecture with ResNet-101 [24] as backbone.

6.1. Pascal VOC2012

Following previous works [56, 42, 43, 4], we design three main experiments adding one class (19-1), five classes at once (15-5) and five classes sequentially (15-1) added in alphabetical order. In Table 1 we report comprehensive results on the three experimental protocols defined in Section 3.1. Results are averaged for mIoU of classes in the base step (old), for classes in the incremental steps (new) and for all classes, and are reported at the end of all the incremental steps. For [4] we also report the original results in their paper (denoted with MiB), that uses a different backbone (thus different pre-trained model) and batch size.

We can appreciate forgetting of previous classes and intransigence in learning new ones even when adding as little as one class (the tv/monitor class is added) in the scenario 19-1. FT always leads to the worst mIoU in terms of old, new and all classes. Incremental methods designed for semantic segmentation allow for a stable improvement across the experimental protocols, in particular MiB, that is specifically targeted to solve the disjoint and the overlapped scenarios, while CIL and ILT encounter difficulties in the overlapped scenario. Also LwF allows for a good improvement while its single-headed version has lower performance in this scenario. Our method (SDR) significantly outperforms all the competitors in the disjoint and overlapped scenarios (with a gap of more than 3% against the best competing approach in the disjoint setup), while in the sequential setup the gap is smaller. Further adding on top of our method the MiB framework (i.e., unbiased cross entropy, knowledge distillation and classifier initialization), which we regard as the current state-of-the-art approach for class incremental semantic segmentation, the results increase on all the scenarios, showing that proposed techniques are complementary with respect to previous schemes.

When moving to the addition of 5 classes at once (i.e., potted plant, sheep, sofa, train, tv/monitor) we immediately notice an overall increased drop of performance of all compared methods, especially in disjoint and overlapped protocols, due to the increased domain shift occurring when adding more classes at once with very variable content. In this and in the following scenario, indeed, we are adding to the model classes belonging to different macroscopic groups, according to [16], which are responsible for a variegated distribution: three indoor classes (potted plant, sofa and tv/monitor), one animal class (sheep) and one vehicle class (train). All compared methods obtain a relevant improvement with respect to FT but are always surpassed by SDR, which in particular outrun the best competing method (MiB) by more than 20% in the disjoint scenario.

In the final scenario we add the last 5 classes sequentially in 5 consecutive learning steps. This approach leads to the largest accuracy drop being the model exposed to a reiterated addition of single classes, which are also coming from different semantic contexts. In the sequential scenario LwF and MiB (which is designed for background shift) show poor final accuracy. ILT and CIL, instead, show results comparable to our approach. In the disjoint and in the overlapped scenarios all the methods heavily suffer from the semantic shift undergone by the background class: LwF (both versions) and ILT have poor performance in these scenar-
Table 1. mIoU on multiple incremental scenarios and protocols on VOC2012. Best in **bold**, runner-up *underlined*. †: results from [4].

<table>
<thead>
<tr>
<th>Method</th>
<th>sequential</th>
<th>19-1</th>
<th>15-5</th>
<th>100-50</th>
<th>50-50</th>
<th>100-10</th>
<th>15-5</th>
<th>50-50</th>
<th>offline</th>
</tr>
</thead>
<tbody>
<tr>
<td>FT</td>
<td>63.4</td>
<td>21.2</td>
<td>61.4</td>
<td>35.2</td>
<td>13.2</td>
<td>34.2</td>
<td>34.7</td>
<td>14.9</td>
<td>33.8</td>
</tr>
<tr>
<td>LwF-FT</td>
<td>67.2</td>
<td>26.4</td>
<td>65.8</td>
<td>28.3</td>
<td>25.0</td>
<td>62.6</td>
<td>23.6</td>
<td>60.5</td>
<td>68.0</td>
</tr>
<tr>
<td>ILT [42]</td>
<td>64.1</td>
<td>22.8</td>
<td>62.6</td>
<td>18.1</td>
<td>60.5</td>
<td>35.1</td>
<td>13.8</td>
<td>34.1</td>
<td>63.8</td>
</tr>
<tr>
<td>MiB [4]</td>
<td>68.2</td>
<td>31.9</td>
<td>66.7</td>
<td>26.0</td>
<td>65.1</td>
<td>69.6</td>
<td>23.8</td>
<td>67.4</td>
<td>70.6</td>
</tr>
<tr>
<td>CIL [33]</td>
<td>64.1</td>
<td>22.8</td>
<td>62.6</td>
<td>18.1</td>
<td>60.5</td>
<td>35.1</td>
<td>13.8</td>
<td>34.1</td>
<td>63.8</td>
</tr>
<tr>
<td>MiB-MC</td>
<td>49.2</td>
<td>9.9</td>
<td>46.9</td>
<td>1.0</td>
<td>36.7</td>
<td>37.1</td>
<td>2.3</td>
<td>35.4</td>
<td>70.6</td>
</tr>
<tr>
<td>IL T [42]</td>
<td>64.1</td>
<td>22.8</td>
<td>62.6</td>
<td>18.1</td>
<td>60.5</td>
<td>35.1</td>
<td>13.8</td>
<td>34.1</td>
<td>63.8</td>
</tr>
<tr>
<td>MiB [4]</td>
<td>68.2</td>
<td>31.9</td>
<td>66.7</td>
<td>26.0</td>
<td>65.1</td>
<td>69.6</td>
<td>23.8</td>
<td>67.4</td>
<td>70.6</td>
</tr>
<tr>
<td>MiB†</td>
<td>68.2</td>
<td>31.9</td>
<td>66.7</td>
<td>26.0</td>
<td>65.1</td>
<td>69.6</td>
<td>23.8</td>
<td>67.4</td>
<td>70.6</td>
</tr>
<tr>
<td>MiB†</td>
<td>68.2</td>
<td>31.9</td>
<td>66.7</td>
<td>26.0</td>
<td>65.1</td>
<td>69.6</td>
<td>23.8</td>
<td>67.4</td>
<td>70.6</td>
</tr>
<tr>
<td>SDR (ours)</td>
<td>70.6</td>
<td>48.9</td>
<td>68.5</td>
<td>70.9</td>
<td>31.4</td>
<td>68.9</td>
<td>71.3</td>
<td>23.4</td>
<td>69.0</td>
</tr>
<tr>
<td>SDR + MiB</td>
<td>70.6</td>
<td>48.9</td>
<td>68.5</td>
<td>70.9</td>
<td>31.4</td>
<td>68.9</td>
<td>71.3</td>
<td>23.4</td>
<td>69.0</td>
</tr>
<tr>
<td>offline</td>
<td>75.5</td>
<td>75.5</td>
<td>75.5</td>
<td>75.5</td>
<td>75.5</td>
<td>75.5</td>
<td>75.5</td>
<td>75.5</td>
<td>75.5</td>
</tr>
</tbody>
</table>

Visual results for each scenario in the disjoint protocol are shown in the first three rows of Fig. 3, where our method is compared against all the competitors consistently obtaining better segmentation maps. For example, our method does not mislead the bus windows with tv/monitor instances in row 1 differently from several competitors (which are more biased toward predicting the novel class), and it is the only one able to distinguish the bus in row 2 and the car in row 3 from the similar-looking train class. Here, train is added in the incremental step causing catastrophic forgetting of similar classes in competing approaches.
on the Pascal dataset in the challenging 15-1 scenario. As already noticed, FT leads to a great degradation of mIoU. Early continual semantic segmentation approaches use a classical output-level knowledge distillation [42, 43, 33] which show discrete benefits boosting the mIoU by almost 20%. Each component of the approach significantly contributes to the final mIoU providing non-overlapping and mutual benefits. Matching prototypes, sparsifying features vectors and constraining them via the contrastive objective regularize the latent space bringing large improvements on both old and new classes. We observe that also the contrastive loss brings a significant contribution if applied alone improving the mIoU of 13.5%. Introducing standard output-level knowledge distillation on top increases the accuracy on old classes mainly, and its unbiased version prevents forgetting even more accounting for the background shift across the incremental learning steps.

Finally, we show that two of the proposed approaches (namely, sparsity and contrastive learning) may be beneficial also for the more general case of standard (i.e., non-incremental) semantic segmentation. Hence, we conduct some additional experiments on Pascal VOC2012 and ADE20K, reported in Table 4, showing the clear benefit of the two components in this setup. On both datasets the outcome is consistent, gaining 0.9% and 1% respectively, even starting from an architecture (i.e., Deeplab-v3+) which is already state of the art.

7. Conclusion

In this paper we presented some latent representation shaping techniques to prevent forgetting in continual semantic segmentation. In particular, the proposed constraints on the latent space regularize the learning process reducing forgetting whilst simultaneously improving the recognition of novel classes. A prototypes matching constraint enforces latent space consistency on old classes, a features sparsification objective reduces the number of active channels limiting cross-talk between features of different classes, and contrastive learning clusters features according to their semantic while tearing apart those of different classes. Our contribution if applied alone improving the mIoU of 13.5%. Introducing standard output-level knowledge distillation on top increases the accuracy on old classes mainly, and its unbiased version prevents forgetting even more accounting for the background shift across the incremental learning steps.

### Table 2. mIoU over multiple incremental scenarios on disjoint setup of ADE20K. Best in **bold**, runner-up underlined.

<table>
<thead>
<tr>
<th>Method</th>
<th>100-50</th>
<th>old</th>
<th>new</th>
<th>all</th>
<th>100-10</th>
<th>old</th>
<th>new</th>
<th>all</th>
<th>50-50</th>
<th>old</th>
<th>new</th>
<th>all</th>
<th>50-50</th>
<th>old</th>
<th>new</th>
<th>all</th>
<th>50-50</th>
<th>old</th>
<th>new</th>
<th>all</th>
<th>50-50</th>
</tr>
</thead>
<tbody>
<tr>
<td>FT</td>
<td>0.0</td>
<td>22.5</td>
<td>7.5</td>
<td></td>
<td>0.0</td>
<td>2.5</td>
<td>0.8</td>
<td>13.9</td>
<td>12.0</td>
<td>12.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LwF [36]</td>
<td>25.0</td>
<td>23.9</td>
<td>24.6</td>
<td></td>
<td>3.4</td>
<td>5.6</td>
<td>5.5</td>
<td>32.7</td>
<td>22.9</td>
<td>26.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LwF-MC [51]</td>
<td>8.6</td>
<td>0.0</td>
<td>5.8</td>
<td></td>
<td>0.0</td>
<td>0.9</td>
<td>0.3</td>
<td>2.8</td>
<td>0.5</td>
<td>1.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ILL [42]</td>
<td>27.2</td>
<td>21.7</td>
<td>25.4</td>
<td></td>
<td>0.0</td>
<td>0.2</td>
<td>0.8</td>
<td>41.9</td>
<td>21.1</td>
<td>28.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CL [33]</td>
<td>0.0</td>
<td>22.5</td>
<td>7.5</td>
<td></td>
<td>0.0</td>
<td>2.0</td>
<td>0.6</td>
<td>14.0</td>
<td>11.9</td>
<td>12.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MiB [4]</td>
<td>37.6</td>
<td>24.8</td>
<td>33.2</td>
<td></td>
<td>21.0</td>
<td>5.3</td>
<td>15.8</td>
<td>39.1</td>
<td>22.6</td>
<td>28.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDR (ours)</td>
<td>37.4</td>
<td>24.8</td>
<td>33.2</td>
<td><strong>28.9</strong></td>
<td><strong>7.4</strong></td>
<td><strong>21.7</strong></td>
<td>40.9</td>
<td>23.8</td>
<td><strong>29.5</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SDR+MiB</td>
<td>37.5</td>
<td><strong>25.5</strong></td>
<td><strong>33.5</strong></td>
<td></td>
<td><strong>28.9</strong></td>
<td>11.7</td>
<td><strong>23.2</strong></td>
<td>42.9</td>
<td><strong>25.4</strong></td>
<td><strong>31.3</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>offline</td>
<td>43.9</td>
<td>27.2</td>
<td>38.3</td>
<td></td>
<td>43.9</td>
<td>27.2</td>
<td>38.3</td>
<td>50.9</td>
<td>32.1</td>
<td>38.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 3. Ablation on disjoint VOC2012 15-1 in terms of mIoU.

<table>
<thead>
<tr>
<th>$L_{ce}$</th>
<th>$L_{pm}$</th>
<th>$L_{sp}$</th>
<th>$L_{cl}$</th>
<th>$L_{kd}$</th>
<th>old</th>
<th>new</th>
<th>all</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.8</td>
<td>4.9</td>
<td>5.6</td>
<td>76.3</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>30.0</td>
<td>11.0</td>
<td>25.4</td>
<td>75.8</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>18.7</td>
<td>9.0</td>
<td>16.4</td>
<td>75.8</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>40.4</td>
<td>12.9</td>
<td>33.9</td>
<td>75.8</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>41.0</td>
<td>13.2</td>
<td>34.4</td>
<td>75.8</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>50.0</td>
<td><strong>15.9</strong></td>
<td>41.9</td>
<td>75.8</td>
</tr>
</tbody>
</table>

### Table 4. Standard (non-incremental) semantic segmentation.

<table>
<thead>
<tr>
<th>$L_{ce}$</th>
<th>$L_{pm}$</th>
<th>$L_{sp}$</th>
<th>$L_{cl}$</th>
<th>mIoU-VOC2012</th>
<th>mIoU-ADE20K</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>75.4</td>
<td>38.3</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>75.8</td>
<td>38.7</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>75.8</td>
<td>38.8</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>76.3</td>
<td><strong>39.3</strong></td>
</tr>
</tbody>
</table>

### 6.2. ADE20K

Following [4] we split the dataset into disjoint image sets with the only constraint that a minimum number of images (i.e., 50) have labeled pixels on $C_k$. Classes are ordered according to [76]. In this comparison we report the same competing methods of Section 6.1. The scenarios we consider are the addition of the last 50 classes at once (100-50), of the last 50 classes 10 at a time (100-10) and of the last 100 classes in 2 steps of 50 classes (50-50). The results are summarized in Table 2, where we can appreciate that the proposed approach outperforms competitors in every scenario, in particular with a larger gain when multiple incremental steps are performed. When adding 50 classes at a time LwF-MC and CL achieve low results and are outperformed by the other competitors (i.e., LwF, ILL and MiB), which in turn are always consistently surpassed by our framework. In the scenario 100-10, instead, all competing approaches (except for MiB) are unable to provide useful outputs leading to extremely low results, while our method stands out from competitors outperforming also MiB by a good margin. Visual results for each scenario are reported as last rows of Fig. 3, which confirm our considerations showing how SDR produces less noisy predictions and does not overestimate the background as some competitors.

### 6.3. Ablation Study

To evaluate the effect of each component, we report an ablation analysis in Table 3 on the Pascal dataset in the challenging 15-1 scenario. As already noticed, FT leads to a
References


1122


