

Is Continual Learning Ready for Real-world Challenges?

Theodora Kontogianni¹ Yuanwen Yue¹ Siyu Tang¹ Konrad Schindler¹

Abstract

Despite continual learning’s long and well-established academic history its application in real-world scenarios remains rather limited. This paper contends that this gap is attributable to a misalignment between the actual challenges of continual learning and the evaluation protocols in use, rendering proposed solutions ineffective for addressing the complexities of real-world setups. We validate our hypothesis and assess progress to date, using a new 3D semantic segmentation benchmark, OCL-3DSS. We investigate various continual learning schemes from the literature by utilizing more realistic protocols that necessitate online and continual learning for dynamic, real-world scenarios (*e.g.*, in robotics and 3D vision applications). The outcomes are sobering: all considered methods perform poorly, significantly deviating from the upper bound of joint offline training. This raises questions about the applicability of existing methods in realistic settings. Our paper aims to initiate a paradigm shift, advocating for the adoption of continual learning methods through new experimental protocols that better emulate real-world conditions to facilitate breakthroughs in the field.

1. Introduction

Continual learning (CL) (Ring et al., 1994) is one of the fundamental machine learning fields. Its methodology emerged as a solution for incorporating new knowledge in existing models to adapt to new data distribution such as new labels and tasks (Ring et al., 1994). Furthermore, in scenarios where re-training a model from scratch with the entire dataset is impractical or resource-intensive (*e.g.* foundation models), incremental updates on new data can make the process more efficient.

Despite continual learning’s well-established academic history, spanning decades and tasks such as image classifica-

tion (Aljundi et al., 2018; Kirkpatrick et al., 2017; Li & Hoiem, 2017; Chaudhry et al., 2019; Lopez-Paz & Ranzato, 2017), 2D semantic segmentation (Michieli & Zanuttigh, 2019; Cermelli et al., 2020; Maracani et al., 2021), and recently, 3D semantic segmentation (Camuffo & Milani, 2023; Yang et al., 2023), its practical application in real-world scenarios remains limited (Gonzalez et al., 2020). This paper contends that this gap is attributable to a misalignment between the actual challenges of continual learning and the evaluation protocols in use, rendering proposed solutions ineffective for addressing the complexities of real-world setups.

This paper underscores the importance of recognizing the limitations of current evaluation protocols and advocates for exploring CL methods through new experimental protocols that mirror real-world setups. *The results are surprising: All examined methods exhibit significant failure with near-zero intersection-over-union (IoU) even in medium-sized task sequences (20 tasks) and fall considerably short of the upper bound achievable by joint offline training.*

While CL has traditionally focused on offline image classification (Aljundi et al., 2018; Kirkpatrick et al., 2017; Li & Hoiem, 2017; Chaudhry et al., 2019; Lopez-Paz & Ranzato, 2017), the demand for foundation models in autonomous agents and robotics, capable of dynamic adaptation in real-time 3D environments underscores 3D semantic segmentation as a more suitable task for benchmarking continual learning methods.

Regardless of the task, CL methods adhere to similar evaluation scenarios. These scenarios typically involve (i) multiple training epochs over the current task, allowing for the repetitive shuffling of data to facilitate task learning but not well-suited for streaming data. Additionally, (ii) reliance on large pre-trained models, while introducing only a few additional tasks in a CL scenario, often leading to trivial solutions. Lastly, (iii) a preference for a *disjoint* scenario, where input data (pixels or 3D points) only from previous and current classes are included, an unrealistic assumption, particularly in the context of autonomous agents.

To address some of these limitations, there has been recently a growing interest in Online Continual Learning (OCL) (Ghunaim et al., 2023) that studies the problem of learning from an online stream of data and tasks. Such a

¹ETH Zurich, Switzerland. Correspondence to: Theodora Kontogianni <kontogianni.theodora@gmail.com>.

setup is well-suited for dynamic environments where the learning process is ongoing such as in robotics but OCL methods primarily focus on image classification, as is common in most continual learning studies, without delving into other tasks like semantic segmentation.

In a similar spirit, our setup features a single online data stream. At each time step, a small batch of data arrives, providing the model with an opportunity for learning before transitioning to the next incoming set of data (*single-pass*). Training unfolds over a prolonged sequence of tasks (*long task sequence*), where all classes are learned incrementally, starting immediately from the first task without any batch pre-training. Starting with a randomly initialized model, rather than one pre-trained on a set of tasks, may initially seem less realistic. However, it allows for the simulation of a more extended sequence of tasks in practice. Additionally, in contrast to previous methods emphasizing the *disjoint* scenario, we prioritize the *overlapped* scenario. Here, data points from future classes are also accessible, although without their corresponding ground truth.

To study online continual learning in 3D semantic segmentation, we modified various well-known 3D semantic segmentation datasets, ScanNet (Dai et al., 2017a), S3DIS (Armeni et al., 2016) and SemanticKITTI (Behley et al., 2019) to suit our online continual learning framework and used them to evaluate the performance of numerous widely-used continual learning approaches. Furthermore, inspired by recent approaches for continual learning in images we evaluated the use of a (i) vision-language model and a (ii) transformer decoder.

CL suffers from the pervasive issue known as *catastrophic forgetting* (French, 1999; McCloskey & Cohen, 1989). This refers to the neural networks' tendency to abruptly and entirely erase previously acquired knowledge when learning new information. Online continual learning, which assumes single-pass data, is strictly harder than offline, due to the combined challenges of catastrophic forgetting and underfitting within a single training epoch.

Due to continual learning's long history, there are several works on continual learning from where we select a combination of seminal and state-of-the-art methods to fairly compare in our OCL-3DSS setup: MAS (Aljundi et al., 2018) as a regularization-based approach, LwF (Li & Hoiem, 2017) as a distillation-based and ER (Chaudhry et al., 2019) as a replay-based approach. Furthermore, we assessed the method proposed by Yang et al. (2023), which currently stands as the sole existing approach in continual learning for 3D semantic segmentation. The method primarily relies on the utilization of confident pseudolabels, as is common in many other CL methods. We additionally incorporated and evaluated language guidance and different backbone architectures.

We observe that all methods exhibit comparable results when the number of tasks is small (≤ 5), the common setup in existing benchmarks. However, their performance sharply declines to *almost zero* after a few more tasks even for the recently popular methods with language guidance and transformer architecture. Notably, the replay-based method ER (Chaudhry et al., 2019) stands out as the sole approach to maintain competitiveness in an OCL-3DSS scenario. However, with a mIoU of 42.1% (20 tasks) and utilizing 33% of the dataset in memory for replay, it still lags significantly behind the 66.4% achieved on ScanNet by joint training.

Given the streaming nature of the task, we noticed that many classes are difficult to learn in an online continual learning setup without constant repetition of the class samples as in joint training. *Positive backward* and *positive forward transfer* has not been witnessed with the exception of positive backward transfer for the replay methods as it is expected when replaying previous samples. While replay-based methods may provide a practical solution, we contend that they do not fundamentally address the core aspects of continual learning and adaptation to new experiences. Detailed experiments supporting our assertion are presented in the subsequent sections of the paper.

We encourage future works to adopt our experimental protocol for assessing 2D and 3D semantic segmentation in a continual learning framework. Despite the challenging nature of our real-life setup, where most continual learning methods produce unfavorable results, we believe this is the way toward significant advancements in the field.

2. Related Work

2.1. Offline Continual Learning

The simplest and most common methods in continual learning have been focusing on two main approaches: replay-based and regularization-based methods.

Replay-based methods store input samples or representations of the current task while the model learns from them and later replays them while the model learns new tasks (Lopez-Paz & Ranzato, 2017; Maracani et al., 2021; Chaudhry et al., 2019). Replay-based methods overcome catastrophic forgetting by simply re-training on the stored data of tasks already experienced so far. Some works store past experiences based on knowledge distillation (Hinton et al., 2015) especially when privacy is a concern but the simple storing of a set of raw samples from previous tasks (Chaudhry et al., 2019; Lopez-Paz & Ranzato, 2017) works remarkably well. Replay-based methods are consistently the best-performing ones in continual learning. However, they are bound by the size of the replay buffer and the re-training time with every additional task. It could be argued that these approaches don't fundamentally tackle con-

tinual learning but rather represent an alternative manifestation of joint training.

Regularization Methods. Several previous benchmarks on continual learning impose restrictions by prohibiting access to previously seen data. The rationale behind such constraints is often attributed to concerns related to storage space and privacy. The majority of these methods concentrate on addressing the issue of catastrophic forgetting through regularization, imposing penalties during training based on factors such as the importance of weights for specific tasks (Aljundi et al., 2018; Kirkpatrick et al., 2017). The objective is to identify crucial parameters for each task, restricting their alteration in subsequent tasks according to their performance. Alternatively, some approaches involve knowledge distillation from a prior model (Li & Hoiem, 2017). Notably, these methods operate without the need to store previously labeled examples. While they exhibit satisfactory performance in small datasets and straightforward classification tasks, their efficacy diminishes in challenging and complex scenarios (Ghunaim et al., 2023), as corroborated by our experiments.

Key Issues. The majority of CL evaluations are conducted in small-scale datasets on classification tasks. Furthermore, traditional continual learning methods start with a pre-trained model on a large subset of the tasks and incrementally introduce a handful of new tasks (Fig. 7, Appendix). They allow the model to be trained over multiple epochs on each task with repeated shuffling of data and where the data distribution of each task is stationary.

2.2. Online Continual Learning

To alleviate these limitations, recently OCL methods started focusing on the cases where data arrive one tiny batch at a time and require the model to learn from a single pass of data (Ghunaim et al., 2023).

Key Issues. Image classification remains the sole focus of all these methods as in most CL studies. In contrast, we introduce online learning for 3D semantic segmentation, a more pragmatic setup for autonomous agent tasks.

2.3. Continual Learning in Semantic Segmentation

Recently, increased attention has been devoted to continual learning for semantic segmentation in 2D images (Michieli & Zanuttigh, 2019; Cermelli et al., 2020). The problem was first introduced in (Michieli & Zanuttigh, 2019). Regularization-based PLOP (Cermelli et al., 2020) alleviates forgetting of old knowledge by distilling multi-scale features. Replay-based RECALL (Maracani et al., 2021) obtains more additional data via GAN or web. Recently (Camuffo & Milani, 2023) has presented a first attempt at using CL for semantic segmentation on LiDAR point clouds in outdoor scenes.

Key Issues. All approaches addressing CL in Semantic

Segmentation share the same foundational assumptions as traditional offline CL methods. These involve initiating with a pre-trained model on a substantial subset of tasks and gradually incorporating a few new tasks. Training occurs over multiple epochs for each task, allowing the repeated shuffling of data. The assumption is that the data distribution for each task remains stationary and is drawn from an i.i.d. distribution. *In contrast, our approach introduces CL in 3D Semantic Segmentation by (i) adopting a fully incremental online setting and (ii) deviating from the i.i.d. assumption, making it more suitable for tasks involving autonomous agents.*

3. Problem Formulation

Before we introduce our Online Continual Learning for 3D Semantic Segmentation (OCL-3DSS) setup (Sec. 3.3), we present the setup of semantic segmentation in 3D point clouds (Sec. 3.1) and offline continual learning (Sec. 3.2).

3.1. Standard 3D Semantic Segmentation

Let $x \in \mathbb{R}^{N \times C_{in}}$ be a 3D point cloud consisting of N points and C_{in} input channels (typically $C_{in} = 3$ for XYZ coordinates or $C_{in} = 6$ for XYZ and RGB if color information is available). The segmentation map, denoted as $y \in \mathbb{R}^{N \times |\mathcal{C}|}$, represents semantic classes within the point cloud, where \mathcal{C} is the set of semantic classes.

Given a training set, the objective of semantic segmentation is to learn a model $f_\theta(x) : X \rightarrow Y$ that maps the input space X to a set of class probability vectors $X \rightarrow \mathbb{R}^{N \times |\mathcal{C}|}$. The final prediction for each 3D point is determined by $y_{pred} = \underset{c \in \mathcal{C}}{\operatorname{argmax}} f_\theta(x)[i, c]_{i=1}^N$, where $f_\theta(x)[i, c]$ represents the probability of 3D point i belonging to class c .

3.2. Offline Continual Learning

In contrast to conventional training approaches, continual learning (Ring et al., 1994) involves training a machine learning model on a sequential stream of data from various tasks. Instead of learning from the entire dataset $T \subset X \times Y$ at once, learning occurs in multiple steps denoted as $t = 1, \dots, T$, where only a subset of the classes $\mathcal{C}_t \subset \mathcal{C}$ is available at each step t . This new set of classes $\mathcal{C}_t \cap \mathcal{C}_{1 \dots t-1} = \emptyset$ is used as the ground truth for updating the model parameters. The model is expected to perform optimally not only on the current task t but also to predict and perform well on all classes learned up to that point,

A key challenge in continual learning is the occurrence of *catastrophic forgetting* (French, 1999; McCloskey & Cohen, 1989). This phenomenon arises when a network, trained with data from a new distribution corresponding to task t , tends to *forget* previously acquired knowledge,

resulting in a substantial decline in performance for tasks $1, \dots, t - 1$. From a broader CL perspective, this challenge reflects the *stability-plasticity* dilemma (Grossberg & Grossberg, 1982), where stability entails preserving past knowledge, while plasticity involves the ability to quickly adapt to new information.

3.3. Towards A Realistic Evaluation

Continual Semantic Segmentation (CSS) aims to train a model f_θ over T steps, where at each step t , the model encounters a dataset $D^t = \{x^t, \tilde{y}^t\}$. Here, x^t represents the input image or point cloud, and \tilde{y}^t denotes the ground truth segmentation at time $t \in [1, \dots, T]$. The segmentation map at each step includes only the current classes \mathcal{C}_t , with labels from previous steps $\mathcal{C}_{1:t-1}$ and future steps $\mathcal{C}_{t+1:T}$ collapsed into a *background* class or ignored. In the common *disjoint* setup during the current task t , input pixels or points i corresponding to future tasks x_i^t with $\tilde{y}_i^{t+1:T}$ are excluded. Nevertheless, the model at step t is expected to predict all classes encountered over time, denoted as $\mathcal{C}_{1:t}$. The semantic segmentation task at time t is formulated as:

$$\theta_t = \operatorname{argmin}_{\theta_t} \mathbb{E}_{(x^t, \tilde{y}^t)} \mathcal{L}(f_\theta(x^t), \tilde{y}^t)$$

Online Continual 3D Semantic Segmentation. To establish a foundation for subsequent discussions, we introduce a straightforward continual learning process for sequential fine-tuning on a stream of 3D semantic segmentation tasks $(1, 2, \dots)$. Starting with a randomly initialized model f_{θ_0} , we iteratively adapt to models f_{θ_t} by incorporating \tilde{y}^t into the current model $f_{\theta_{t-1}}$.

In contrast to prior works, we define our continual learning scenario in a more challenging setup, introducing key differences from previous setups in continual semantic segmentation (CSS): (i) The model, f_{θ_0} , is initialized with random weights, implying the absence of old tasks, only previous ones. (ii) In $D^t = \{x^t, \tilde{y}^t\}$, where $t \in \{\{1\}, \dots, \{T\}\}$ and t is a singleton set containing the labels of a single class. (iii) The number of tasks, T , is notably longer. (iv) Each dataset D^t is processed only once.

4. Proposed Benchmark

4.1. A Closer Look At Existing Protocols

In this section, we reflect on the existing evaluation protocols (Yang et al., 2023) and observe that they operate under the common disjoint setting of 2D class incremental segmentation, where the incremental training includes only the old and current classes of the point cloud, excluding the future classes. We find that existing setups adopt the two-set paradigm, one for $\mathcal{C}_{\text{base}}$ and one for $\mathcal{C}_{\text{novel}}$ classes. $\mathcal{C}_{\text{novel}}$ contains only 5, 3, and 1 new classes for both ScanNet (Dai et al., 2017b) and S3DIS (Armeni et al., 2016).

To put these numbers into perspective, in the case of

$\mathcal{C}_{\text{novel}} = 1$ in ScanNet, a pre-trained model with joint training for $\mathcal{C}_{\text{base}} = 19$ classes is updated with a single class. Classes are also split into $\mathcal{C}_{\text{base}}$ and $\mathcal{C}_{\text{novel}}$ based only on the original class label order of the dataset (S_0) or alphabetically (S_1), without taking into consideration the difficulty of each class. Furthermore, 3D semantic segmentation has witnessed tremendous improvement based on sparse convolutional architectures (Choy et al., 2019) and transformers (Schult et al., 2023), which have enhanced the joint training upper bound even further. Notably, these architectures have not been considered in existing protocols.

We understand that the existing protocols in continual learning for 3DSS reflect existing 2D CSS benchmarks however we propose a more realistic setup that can reflect the progress in continual learning more accurately. Existing 3D semantic segmentation datasets can be modified to our OCL-3DSS protocol. To that end we leverage three popular public 3D semantic segmentation datasets: (i) ScanNet (Dai et al., 2017b), (ii) S3DIS (Armeni et al., 2016) and (iii) SemanticKITTI (Behley et al., 2019)) to validate our evaluation setup.

4.2. Datasets

We modify the following datasets for online continual 3D semantic segmentation:

ScanNet (Dai et al., 2017b) is one of the largest real-world 3D datasets on the task of 3D semantic segmentation. It contains 1210 training scenes and 312 scenes for validation with 20 semantic labels. For training and validation, we follow the standard split for 3D semantic segmentation. We split the training set into 20 tasks, each one of them representing the learning of one label. Therefore, even though each 3D scene remains unchanged we use only the ground truth labels of the current class. We would like to process each scene a single time in a true online setup however since ScanNet is a highly imbalanced dataset (see Fig. 8 in Appendix) some object classes are present only in a few scenes. To alleviate the online learning problem and disentangle the task difficulty from the data availability we re-use scenes of the underrepresented classes by sampling with replacement until we have the maximum of 1201 scenes for each class. However, please note that this does not even the number of points labeled for each task (for example classes like *wall* remain one of the most represented ones making the learning still easier). After learning each class t , we evaluate all classes up to and including t on the test set. When we reach class 20, the evaluation is identical to that in joint training. This protocol stands in direct contrast to existing setups, such as 15 – 5, 17 – 3, and 19 – 1 (Yang et al., 2023). Our setup can be denoted as $1 - 1 - 1 \cdots 1 = 1 \times 20$. Notably, we not only incrementally add classes, but this approach also results in a much longer sequence of tasks.

S3DIS (Armeni et al., 2016) contains point clouds of 272 rooms in 6 indoor areas with 13 semantic classes. We use the challenging Area 5 as test and the rest as training in an online continual learning setup as above. We use all classes except *clutter*.

SemanticKITTI (Behley et al., 2019) is an outdoor dataset without RGB information (compared to ScanNet and S3DIS). It is a very big dataset so to align with the online setup we selected sequence 0 for training and sequence 8 for testing. We use 9 classes present in those scenes for our online continual learning setup: *car, road, sidewalk, building, fence, vegetation, trunk, terrain and pole*.

4.3. Task Difficulty Scenarios

Since the order of arrival of each task could influence the performance we organize three different learning scenarios: 1. *standard*, 2. *difficult*→*easy*, 3. *easy*→*difficult*. The *standard* refers to the existing order from the dataset creators, *difficult*→*easy* introduces tasks from the rarest to the most common class based on the number of 3D points of each class and *easy*→*difficult* uses the inverse order.

4.4. Decoder Architectures

We explore two decoder architectures. One is a standard convolutional head that predicts semantic logits. In addition, we adapt the recent Transformer decoders (Strudel et al., 2021; Cheng et al., 2022) for image semantic segmentation to our online continual 3D semantic segmentation. We show our findings are independent from specific decoder architectures.

5. Class Supervision Using Language Models

Recently, prompt tuning has surfaced as an alternative to rehearsal buffers in continual learning for 2D semantic segmentation. Approaches in this domain (Wang et al., 2022b;a; Khan et al., 2023) leverage a pre-trained vision encoder and employ prompt learning to facilitate the continuous learning of tasks, eschewing the need to replay samples from preceding tasks.

Integrating language into continual learning for 3D semantic segmentation presents challenges due to the scarcity of images typically found in such datasets. In the absence of readily available images, our reliance is solely on text prompts for guidance and instruction.

For a given training sample (x, y_t) consisting of a point cloud x with point labels y_t , we express the class name for class t in language as the following prompt:

“A photo of **class name**”

The prompt is input to a pre-trained text encoder to extract

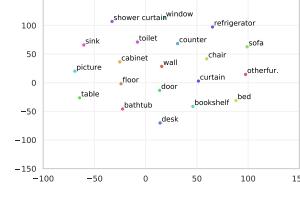


Figure 1. T-SNE of CLIP features on ScanNet (Dai et al., 2017a) classes, evenly distributed in 2D space.

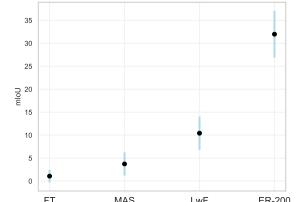


Figure 2. Performance variance on the task ordering in ScanNet.

the feature corresponding to the output token, representing $E_{lang}^t \in \mathbb{R}^C$, the language representation of class t with embedding dimension C .

We aim to constrain the feature encodings E_l^t of class t in the model at layer l to be close to the language representation of the same class. We introduce language guidance through this feature using cosine similarity:

$$S(E_{lang}^t, E_l^t) = \frac{E_{lang}^t \cdot E_l^t}{\|E_{lang}^t\| \cdot \|E_l^t\|} \quad (1)$$

We define $S(E_{lang}^t, E_l^{t'})$ similarly. We optimize the cosine loss to incentivize the model to align the per 3D point features of the network close to the language representation of their respective class E_{lang}^t . Simultaneously, we mitigate the forgetting of previous tasks by keeping the features corresponding to all other classes t' far away from E_{lang}^t . This leverages the fact that the text features are already separable (Fig. 1), thus alleviating forgetting.

Given the task t , the model only has access to the current task labels and no access to labels from other tasks. Therefore, we optimize the following triplet loss based on the cosine similarity described above:

$$L(A, P, N) = -S(A, P) + S(A, N) + \text{margin} \quad (2)$$

The total optimization is defined as:

$$J^t = \sum_{i=1}^M L(A^t, P^i, N^i) \quad (3)$$

Here, we define the anchor point A^t as the feature encoding E_{lang}^t , $P = E_l^t$ as the feature vector of points belonging to class t , and $N = E_l^{t'}$ as the feature vector of points not belonging to class t . We sample an equal number M of points i for both the negative and positive points.

During inference, we assign a label to each point based on the maximum similarity to the language features. This way we require no adaptive decoder and we can add as many classes as needed.

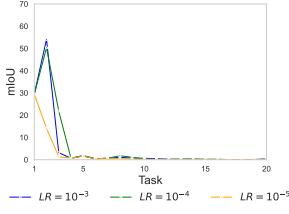


Figure 3. Learning rate in FT
Figure 4. MAS weight does not
has minimal influence.

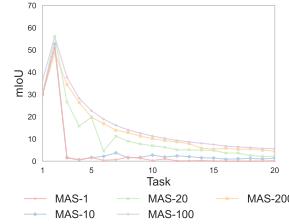


Figure 4. MAS weight does not
significantly impact forgetting.

6. Experiments

Existing Approaches and Baselines. We consider the following state-of-the-art approaches for our empirical studies: MAS (Aljundi et al., 2018), LwF (Li & Hoiem, 2017) and ER (Chaudhry et al., 2019). We choose them as canonical examples for regularization-based, distillation-based and memory replay approaches, discussed in Section 2. LwF regularizes the output of the network in task t based on the outputs of the network of task $t-1$ by always saving the model parameters of the previous task. MAS penalizes the change of the important parameters for the previous tasks basing their importance on the sensitivity of the predicted output to the change in the parameter. ER as in (Ghunaim et al., 2023) performs one gradient step on a batch, half of which is randomly sampled from the memory. We also evaluate Yang et al. (2023) as the only baseline in continual learning for 3D semantic segmentation that combines pseudo-labels created by the model from previous tasks with a distillation loss.

Finally, we also compare with a reference model learned in a traditional semantic segmentation joint training (JT) without continual learning, which may constitute an upper bound. Our lower bound consists of fine-tuning with new task data (FT) without taking any measures to alleviate catastrophic forgetting.

Metrics. We report mean *Intersection over Union* (mIoU) both after the final incremental step and at the intermediate stages. Notably, while many methods demonstrate satisfactory performance for up to three additional tasks, there is a tendency to completely forget earlier tasks in longer sequences. We additionally report:

$$\text{forgetting}_t = \frac{\text{IoU}_t - \text{IoU}_t^T}{\text{IoU}_t} \times 100\%$$

where IoU_t is the current task t performance at step t and IoU_t^T is the performance of task t after all the steps. The smaller the value, the better. Negative values mean improved learning in future tasks.

Implementation Details. During each training step t , a batch consists of 10 3D scenes. In the case of ER, 5 scenes are randomly sampled from memory. This process repeats until all scenes for each task are utilized. To handle the long-tail nature of 3D semantic segmentation, where some

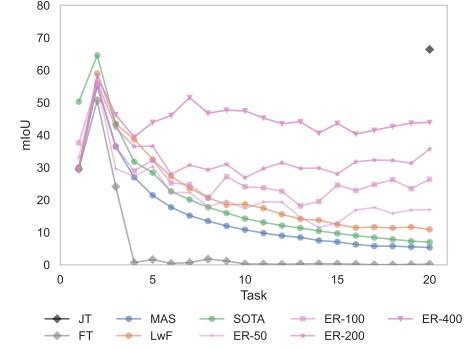


Figure 5. Overall performance of CL methods on our OCL-3DSS setup (ScanNet-v2). All CL methods except ER converge to almost zero by the 20_{th} task. Despite the sizable memory buffer, even ER is still far from JT. ER is more of a practical solution.

classes appear only in a limited number of scenes, we address this by resampling scenes until the count aligns with the maximum available scenes of the most prevalent task.

Backbones. We use the MinkowskiEngine (ME) (Choy et al., 2019) with a 5 cm voxel size as a backbone that we train from scratch starting on task 1. We use SGD with a fixed common learning rate for all tasks. Besides the standard ME decoder, we also modify a Transformer decoder (Strudel et al., 2021; Cheng et al., 2022) to our task. For the language guidance LG-CLIP, we use CLIP (Radford et al., 2021).

Supervision. All methods are fully supervised with all the 3D points with the label of the current class. When language guidance is used supervision requires only 10 points per class per scene and we follow that scenario in all our table entries that report CLIP. The supervision of the other methods is at the level of at least hundreds of points per scene (depending on the class popularity, even thousands).

7. Analysis and Discussion

The challenge: Differences between JT and FT. We first study the impact of incremental fine-tuning, in a relatively long sequence of tasks 20, 12, and 9 for ScanNet, S3DIS and SemanticKITTI respectively. There is a big discrepancy between the Joint Training (JT) of a set of tasks and Finetuning (FT) in each task sequentially as shown in Tables (4, 5, 6) and Fig. 5. For example, as it can be seen in Tab. 4, JT on the 20 classes of the ScanNet dataset with a batch size of 10 for 25 epochs returns an mIoU of 66.4 in the validation set. This can be considered the upper bound for the task on such a backbone. To understand the scale of the catastrophic forgetting issue the sequential training of tasks without any countermeasures to mitigate the catastrophic forgetting yields 0.4 mIoU (Fig. 5). Besides *catastrophic forgetting*, online learning of tasks sequentially encounters an additional challenge: learning with limited data (Tab. 7).

Table 1. OCL-3DSS on ScanNet-val (Dai et al., 2017b) after 3, 5, 10 and 20 (final) tasks. Augmenting the point clouds did not have an influence on other methods besides the FT. Using augmentation for training in cases where the i.i.d assumption does not hold (eg. multiple scans of the same scene) might be comparable to MAS and LwF for few tasks (≤ 5).

Method	mIoU after t tasks			
	3	5	10	20
FT (lower bound)	24.9	3.9	2.1	0.4
FT + Aug.	27.3	17.4	9.1	4.1

Table 2. OCL-3DSS on ScanNet-val (Dai et al., 2017b) after 3, 5, 10 and 20 (final) tasks with *dice loss* instead of cross entropy. Only ER (Chaudhry et al., 2019) improved. The other methods had comparable results (see Tab. 8 in Appendix).

Method	mIoU after t tasks			
	3	5	10	20
ER-200 (Chaudhry et al., 2019)	42.2	36.6	26.8	35.7
ER-200 (Chaudhry et al., 2019) + DiceLoss	47.1	41.7	43.8	36.1

Table 3. OCL-3DSS on ScanNet-val (Dai et al., 2017b) after 3, 5, 10 and 20 (final) tasks with limited use of memory. Just a single stored example from each class performs almost as well as 50. With the additional language guidance, it can surpass the 50 examples in the 5 and 10 classes. The size of the memory matters actually at the 20 classes. So storing several examples per task is important for large task sequences but few examples need to be stored for a moderate number of tasks.

Method	mIoU after t tasks			
	3	5	10	20
ER-50 (Chaudhry et al., 2019)	29.6	30.3	17.4	17.0
ER-2 (Chaudhry et al., 2019)	37.0	29.9	18.3	14.1
ER-1 (Chaudhry et al., 2019)	31.7	24.8	18.5	8.8
ER-1+LG-Clip	30.5	34.1	20.7	8.1

Is OCL purely an optimization problem? Adjusting the learning rate (Fig. 3) does not alleviate forgetting nor does augmentation (Tab. 1). Dice loss (Tab. 2) alleviates the class imbalance and facilitates learning new classes only in ER. **Are regularization methods a solution?** We have evaluated one of the most popular regularization-based methods MAS (Aljundi et al., 2018). MAS has shown good results in classification tasks when using a pre-trained network on ImageNet where new tasks are added (a popular setup in offline CL on classification tasks). However, in our challenging setup of incremental learning from scratch the method performs poorly (Tables 4,5,6 and Fig. 4). Different weights for the MAS loss compared to the segmentation loss do not seem to have a significant effect on the final outcome (Fig. 4). Large weight values just reduce the plasticity. Contrary to (Aljundi et al., 2018), in our case, LwF (Li & Hoiem, 2017) outperforms MAS showing that in our more challenging setup, the existing takeaways from the CL field of image classification do not necessarily hold. It is very important to point out that both methods experience catastrophic forgetting almost after the 3rd task that cannot be mitigated both in ScanNet and S3DIS. In SemanticKITTI (Tab. 6) LwF seems to perform very well.

Is it possible to CL without memory? As shown in Ta-

Table 4. OCL-3DSS on ScanNet-val (Dai et al., 2017b) after 3, 5, 10 and 20 (final) tasks. Best method is ER-400. Most methods offer comparable results up to the 5th task but only replay methods maintain performance after. LG-CLIP is weakly supervised with just 10 3D points per class.

Method	Mem. Size slots · C	Mem./Data (percent)	mIoU after t tasks			
			3	5	10	
FT(lower bound)	-	-	24.9	3.9	2.1	0.4
MAS (Aljundi et al., 2018)	-	-	33.7	22.6	10.8	4.9
LwF (Li & Hoiem, 2017)	-	-	41.2	34.1	17.4	7.8
ER-50 (Chaudhry et al., 2019)	50 · 20	4.1	29.6	30.3	17.4	17.0
ER-200 (Chaudhry et al., 2019)	200 · 20	16.4	42.2	36.6	26.8	35.7
ER-400 (Chaudhry et al., 2019)	400 · 20	32.8	43.6	36.4	39.0	42.1
Yang et al. (2023)	-	-	40.7	24.3	12.1	7.0
LG-Clip-weak sup.	-	-	37.5	18.5	10.4	3.2
JT(upper bound)	-	-	-	-	-	66.4

Table 5. OCL in 3D semantic segmentation on S3DIS-A5 (Armeni et al., 2016) after 3, 6 and 12 (final) tasks.

Method	Mem. Size slots · C	Mem./Data (percent)	mIoU after t tasks		
			3	6	12
FT(lower bound)	-	-	25.1	16.3	9.1
MAS (Aljundi et al., 2018)	-	-	29.5	14.7	8.6
LwF (Li & Hoiem, 2017)	-	-	30.8	8.3	6.4
ER-20 (Chaudhry et al., 2019)	20 · 12	9.8	33.2	17.4	8.6
ER-50 (Chaudhry et al., 2019)	50 · 12	24.5	33.7	18.3	13.7
ER-100 (Chaudhry et al., 2019)	100 · 12	49.0	34.3	19.4	16.5
LG-Clip-weak sup.	-	-	34.2	11.6	4.4
JT(upper bound)	-	-	-	-	65.4

bles 4,5,6 the best performing method is ER (Chaudhry et al., 2019). It is a simple method where half of the training batch consists of samples of previous tasks randomly selected. In our case, we store the raw labels of the 3D points of previous tasks. As anticipated, the size of the memory is a relevant factor, at least up to a certain threshold, especially as the number of tasks increases. It's crucial to highlight that despite storing nearly one-third of our dataset, the performance at the conclusion of 20 tasks reaches, at best, approximately 60% of the upper limit for ScanNet (Dai et al., 2017a) (Tab. 4). Comparable findings are observed in S3DIS (Armeni et al., 2016) (Tab. 5) and SemanticKITTI (Behley et al., 2019) (Tab. 6). The second-best approach is LwF (Li & Hoiem, 2017), which enforces regularization on the new model predictions, compelling them to align closely with the predictions of the model from the previous task. Following closely in performance is the method proposed by Yang et al. (2023), which derives much of its effectiveness from utilizing pseudolabels generated from confident predictions in previous tasks.

Table 6. OCL-3DSS on SemanticKITTI (Behley et al., 2019) after 3, 5 and 9 (final) tasks.

Method	Mem. Size slots · C	Mem./Data (percent)	mIoU after t tasks		
			3	5	9
FT(lower bound)	-	-	6.4	4.9	3.4
MAS (Aljundi et al., 2018)	-	-	7.4	3.3	7.1
LwF (Li & Hoiem, 2017)	-	-	41.1	44.5	49.1
ER-200 (Chaudhry et al., 2019)	200 · 9	4.4	29.0	33.5	48.5
ER-500 (Chaudhry et al., 2019)	500 · 9	11.0	29.0	35.2	52.5
ER-1000 (Chaudhry et al., 2019)	1000 · 9	22.0	31.0	34.4	52.3
LG-Clip-weak sup.	-	-	3.2	8.1	2.2
JT(upper bound)	-	-	-	-	67.4

Table 7. Forgetting per class on ScanNet-val (Dai et al., 2017a). All methods except ER converge to almost zero IoU. Methods like LwF (Li & Hoiem, 2017) and Yang et al. (2023) try to alleviate it but they just remember the populous classes like *wall* and *floor* and fail to learn new ones. With - we represent **underfitting** (zero IoU while learning that class). Negative scores are good → mIoU is increasing with the progression of tasks. Possible only with replay or pseudolabels.

Forgetting↓		wall ↓	floor ↓	cabinet ↑	bed ↓	chair ↓	sofa	table ↓	door	window ↓	bookshelf↑†	picture↑†	counter ↑	desk ↓	curtain ↓	fridge ↓	shcurtain ↓	toilet ↓	sink ↓	bathtub ↑	mIoU ↑
Method																					
FT		100	100	100	99	99	95	100	98	100	100	100	100	100	100	100	100	100	100	0.4	
MAS (Aljundi et al., 2018)		-52	3	72	11	-	52	-	-	-	-	-	-	-	-	-	-	-	-	4.9	
LwF (Li & Hoiem, 2017)		-32	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	7.8	
ER-50 (Chaudhry et al., 2019)		-58	-43	100	-1467	-116	-282	-18	67	-75	-367	100	-67	-47	-225	100	-200	-533	100	-140	17.0
Yang et al. (2023)		-0.3	-4	-	-0.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	7.0	
LG-CLIP-weak sup.		30	52	95	79	96	96	-1	92	95	62	5	-884	-142	-115	-9	46	-200	42	-1	3.2

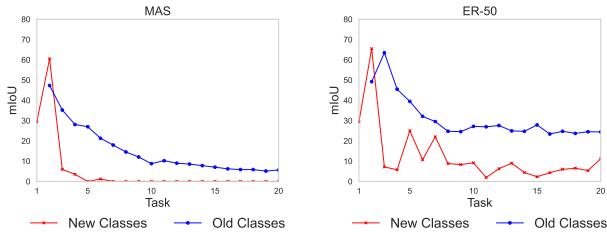


Figure 6. Old vs new class performance with MAS (Aljundi et al., 2018) and ER (Chaudhry et al., 2019). This provides a clearer understanding of the plasticity and rigidity of the method. MAS sacrifices plasticity (learning of new classes) for stability. But even in the best performing ER the learning of the new classes is limited in an online scenario.

However, if objects from the previous class are absent in the scene, the method struggles, resulting in the retention of only ubiquitous classes like walls and floors that are present in every scene (Tab. 7). It is noteworthy to reiterate that both methods experience a significant performance decline after the 5th task, reinforcing our assertion that online CL and CL, in general, necessitate much longer sequences of tasks for a comprehensive evaluation. SemanticKITTI stands out as an exception, given that the majority of its classes are present in every scene. While memory replay methods appear to be at the forefront of continual learning, with discussions extending to the concept of infinite memory (Prabhu et al., 2023), we would like to point out that this is not solving the fundamental issues of learning continuously and adapting with few data but instead reformulated the problem as another joint training setup.

How much memory do we actually need? As seen Tables 4,5,6 and Fig. 5 the memory size is a relevant factor, at least up to a certain threshold, especially as the number of tasks increases. However, in Tab. 3 we see that storing just a single example for each class ER-1 (20 examples in total for ScanNet-v2), is comparable to ER-50 (50 examples per class) in the low task regime of ≤ 5 tasks.

Can large-scale language models help? Inspired by the recent advances in open-world segmentation with the use of image-language models like CLIP (Peng et al., 2023) we co-embedded our 3D point features with text from the CLIP feature space to alleviate catastrophic forgetting since CLIP features are already easily separated (Sec. 5 for de-

tails). Our scenario is much more challenging: we do not use images and the classes for regularization appear separately. But with only 10 points per class for supervision language guidance offers comparable results to the other methods (Tables 4,5,6) that are fully supervised. It can also be used with a very small memory buffer to improve memory replay (Tab. 3).

Can we learn new classes? Our setup presents significant challenges. Firstly, it is highly imbalanced, with many classes appearing in only a few scenes and with few points. Secondly, in our online configuration, the model encounters the data only once. To ensure a fair comparison and prevent performance bias towards classes with more instances, we reused scenes containing underrepresented classes. This approach ensures that our model encounters an equal number of scenes for each class, aligning with the second point of our setup, which aims to simulate a challenging but fair online scenario. Mastering new tasks with such limited data proves to be as challenging as mitigating forgetting, even in the best-performing ER scenarios (Fig. 6, Tab. 7).

Task order influence. The overall performance is impacted by the order the classes appear. Therefore, we conduct our experiments using three distinct class orderings: the original task order from the dataset setup, easy-to-hard, and hard-to-easy. Task difficulty is determined by the number of points of each class (more points → easier task). *Standard* is close to *difficult* and we in general ward against the *easy* as optimistic. We show mean and variance in Fig. 2.

Decoder architecture influence. We empirically find the challenge of OCL-3DSS cannot be solved simply with an advanced Transformer decoder. Details in the Appendix.

8. Conclusion

In this work, we expose the issues of the common practices on continual learning methods. We provide initial attempts towards characterizing the difficulty of continual learning even in moderate-sized task sequences (≥ 5) and learning sequentially tasks with limited data. We recommend future works to evaluate their proposed continual learning methods on such more challenging closer to real-world setups.

References

- Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., and Tuytelaars, T. Memory Aware Synapses: Learning What (Not) To Forget. In *European Conference on Computer Vision (ECCV)*, 2018. 1, 2, 3, 6, 7, 8, 12, 13
- Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., and Savarese, S. 3D Semantic Parsing of Large-Scale Indoor Spaces. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 4, 5, 7, 11, 12, 13
- Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., and Gall, J. SemanticKITTI: A Dataset for Semantic Scene Understanding of Lidar Sequences. In *International Conference on Computer Vision (ICCV)*, 2019. 2, 4, 5, 7, 11, 13
- Camuffo, E. and Milani, S. Continual Learning for LiDAR Semantic Segmentation: Class-Incremental and Coarse-to-Fine Strategies on Sparse Data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2023. 1, 3
- Cermelli, F., Mancini, M., Bulo, S. R., Ricci, E., and Caputo, B. Modeling the Background for Incremental Learning in Semantic Segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 3
- Cermelli, F., Cord, M., and Douillard, A. CoMFormer: Continual Learning in Semantic and Panoptic Segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 12
- Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P., Torr, P., and Ranzato, M. Continual Learning With Tiny Episodic Memories. In *Workshop on Multi-Task and Lifelong Reinforcement Learning*, 2019. 1, 2, 6, 7, 8, 12, 13, 14
- Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., and Girdhar, R. Masked-attention Mask Transformer for Universal Image Segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5, 6
- Choy, C., Gwak, J., and Savarese, S. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4, 6, 12, 13
- Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., and Nießner, M. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017a. 2, 5, 7, 8, 11
- Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., and Nießner, M. Scannet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017b. 4, 7, 12, 13
- Deng, R., Shen, C., Liu, S., Wang, H., and Liu, X. Learning to predict crisp boundaries. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 562–578, 2018. 11
- French, R. M. Catastrophic Forgetting in Connectionist Networks. *Trends in Cognitive Sciences*, 1999. 2, 3
- Ghunaim, Y., Bibi, A., Alhamoud, K., Alfarrar, M., Al Kader Hammoud, H. A., Prabhu, A., Torr, P. H., and Ghanem, B. Real-Time Evaluation in Online Continual Learning: A New Hope. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 3, 6, 13
- Gonzalez, C., Sakas, G., and Mukhopadhyay, A. What is wrong with continual learning in medical image segmentation? *arXiv preprint arXiv:2010.11008*, 2020. 1
- Grossberg, S. and Grossberg, S. How Does a Brain Build a Cognitive Code? *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control*, 1982. 4
- Hinton, G., Vinyals, O., and Dean, J. Distilling the Knowledge in A Neural Network. *arXiv preprint arXiv:1503.02531*, 2015. 2
- Khan, M. G. Z. A., Naeem, M. F., Gool, L. V., Stricker, D., Tombari, F., and Afzal, M. Z. Introducing Language Guidance in Prompt-based Continual Learning. In *International Conference on Computer Vision (ICCV)*, 2023. 5
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming Catastrophic Forgetting in Neural Networks. *Proceedings of the National Academy of Sciences*, 2017. 1, 3
- Li, Z. and Hoiem, D. Learning Without Forgetting. *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2017. 1, 2, 3, 6, 7, 8, 11, 12, 13
- Lopez-Paz, D. and Ranzato, M. Gradient Episodic Memory for Continual Learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 1, 2
- Loshchilov, I. and Hutter, F. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2017. 13
- Maracani, A., Michieli, U., Toldo, M., and Zanuttigh, P. Recall: Replay-Based Continual Learning in Semantic Segmentation. In *International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 3
- Mccloskey, M. and Cohen, N. J. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In *Psychology of learning and motivation*. 1989. 2, 3
- Michieli, U. and Zanuttigh, P. Incremental Learning Techniques for Semantic Segmentation. In *International Conference on Computer Vision (ICCV) Workshops*, 2019. 1, 3
- Peng, S., Genova, K., Jiang, C. M., Tagliasacchi, A., Pollefeys, M., and Funkhouser, T. OpenScene: 3D Scene Understanding with Open Vocabularies. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 8
- Prabhu, A., Cai, Z., Dokania, P., Torr, P., Koltun, V., and Sener, O. Online Continual Learning Without the Storage Constraint. *arXiv preprint arXiv:2305.09253*, 2023. 8
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, 2021. 6, 13

Ring, M. B. et al. Continual Learning in Reinforcement Environments. 1994. 1, 3

Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., and Leibe, B. Mask3D for 3D Semantic Instance Segmentation. In *International Conference on Robotics and Automation (ICRA)*, 2023. 4

Shang, C., Li, H., Meng, F., Wu, Q., Qiu, H., and Wang, L. Incrementer: Transformer for Class-Incremental Semantic Segmentation With Knowledge Distillation Focusing on Old Class. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 12

Strudel, R., Garcia, R., Laptev, I., and Schmid, C. Segmenter: Transformer for semantic segmentation. In *International Conference on Computer Vision (ICCV)*, 2021. 5, 6

Vitter, J. S. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985. 14

Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.-Y., Ren, X., Su, G., Perot, V., Dy, J., et al. Dualprompt: Complementary Prompting for Rehearsal-Free Continual Learning. In *European Conference on Computer Vision (ECCV)*, 2022a. 5

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., and Pfister, T. Earning to Prompt for Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b. 5

Yang, Y., Hayat, M., Jin, Z., Ren, C., and Lei, Y. Geometry and Uncertainty-Aware 3D Point Cloud Class-Incremental Semantic Segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 4, 6, 7, 8, 11, 13

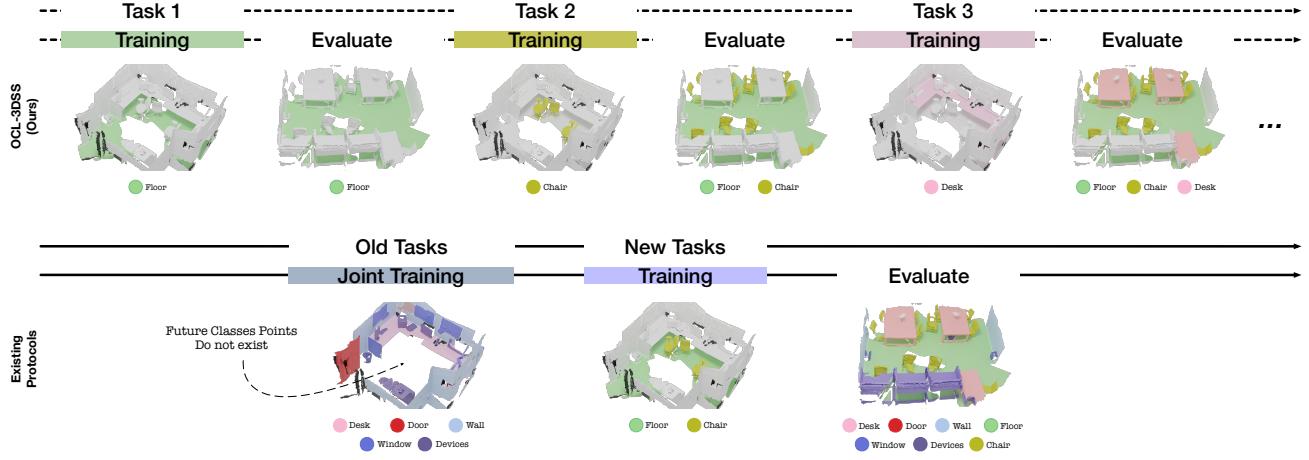


Figure 7. An illustration of our continual learning framework. Our suggested protocol (top row) vs. the existing protocols (bottom row) (i) learn classes incrementally instead of jointly training for the majority of classes (ii) evaluates on long sequences of tasks instead of a two training step approach (iii) contains points belonging to future classes (*floor* and *chair* are missing in 2_{nd} row, 1_{st} col.)

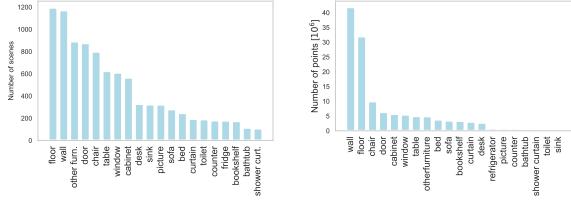


Figure 8. ScanNet-v2 (Dai et al., 2017a). Number of scenes a class is present (left) and number of 3D points labeled for each class (right). The number of points determines the easiness of a task. The right figure represents the task ordering of *easy*→*difficult* scenario described in Sec. 4.3 in the main paper.

A. Dataset Statistics

ScanNet-v2 (Dai et al., 2017a) stands out as one of the most extensive indoor 3D datasets, characterized by a long-tail distribution (see Fig. 8, left). This distribution, reflecting real-world scenarios, is not only evident in the points per class but also in the frequency of scenes featuring an object class. Real-world occurrences of certain objects are less frequent, making both learning and retaining them challenging. This difficulty is compounded by the fact that these objects don't appear in every scene, even unlabeled in different tasks, rendering methods like pseudo labels (Yang et al., 2023) or LwF (Li & Hoiem, 2017) less effective. Due to its alignment with real-world distribution and having the highest number of classes (20), the ScanNet dataset is chosen for all our ablation studies.

SemanticKITTI (Behley et al., 2019), on the other hand, is a large-scale outdoor LiDAR dataset that we employ for training using 4541 scenes from sequence-0. As illustrated in Fig. 9, SemanticKITTI doesn't exhibit a long-tail problem; most labeled outdoor classes are nearly ubiquitous across scenes. This characteristic mitigates issues related to

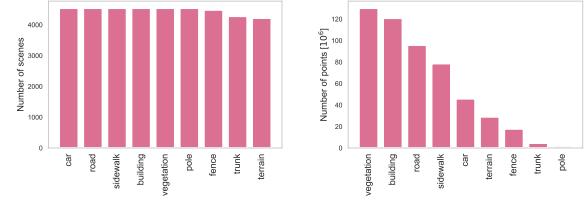


Figure 9. SemanticKITTI (Behley et al., 2019). Distribution of class presence across scenes (left) and the number of labeled 3D points for each class (right). Our SemanticKITTI setup notably lacks the typical long tail problem (left) encountered in online continual learning, as nearly all 9 classes are present in every scene. When benchmarking OCL, it is crucial to consider not only the point count (right) but also the frequency of scenes featuring each object (left).

continual learning, as demonstrated in our evaluation (see Tab. 6 in the main paper). However, it also highlights that certain datasets, despite their vast size in the number of scenes, may not be ideal benchmarks for evaluating continual learning methods.

S3DIS (Armeni et al., 2016) occupies a middle ground between the more diverse datasets of ScanNet and SemanticKITTI, as illustrated in Fig. 10. Our training set encompasses a total of 204 scenes for Areas 1-4,6, with Area 5 reserved for evaluation, following standard practices in the field.

B. Loss Functions

In addition to the conventional cross-entropy loss, we assessed our configuration by incorporating the Dice loss (Deng et al., 2018) (Tab. 8). While the Dice loss was originally crafted to enhance the definition of boundaries, it also proved beneficial in addressing class imbalances within a standard joint training framework. It demon-

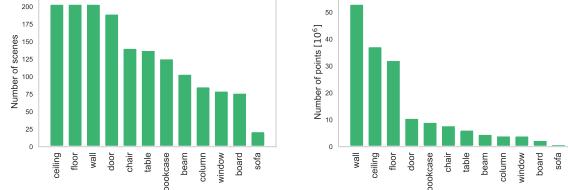


Figure 10. S3DIS (Armeni et al., 2016). Number of scenes a class is present (left) and number of 3D points labeled for each class (right).

Table 8. OCL-3DSS on ScanNet-val (Dai et al., 2017b) after 3, 5, 10 and 20 (final) tasks incorporating an additional *dice loss* alongside cross-entropy. Notably, only ER (Chaudhry et al., 2019) exhibited improvement.

Method	mIoU after t tasks			
	3	5	10	20
FT(lower bound)	24.9	3.9	2.1	0.4
FT + DiceLoss	23.0	1.6	0.7	0.9
MAS (Aljundi et al., 2018)	33.7	22.6	10.8	4.9
MAS (Aljundi et al., 2018) + DiceLoss	34.4	19.2	8.9	4.4
LwF (Li & Hoiem, 2017)	41.2	34.1	17.4	7.8
LwF (Li & Hoiem, 2017) + DiceLoss	31.5	25.8	16.3	7.1
ER-200 (Chaudhry et al., 2019)	42.2	36.6	26.8	35.7
ER-200 (Chaudhry et al., 2019) + DiceLoss	47.1	41.7	43.8	36.1
LG-Clip-weak sup.	37.5	18.5	10.4	3.2
LG-Clip + DiceLoss	25.2	20.9	8.6	5.8
JT (upper bound)	-	-	-	66.4

strates a modest improvement in scenarios akin to joint learning, such as ER (Chaudhry et al., 2019). Nevertheless, it imposes penalties on prevalent, easily identifiable classes, contrasting with other methodologies like LwF (Li & Hoiem, 2017) that rely on them for their performance metrics.

C. The Importance of the Decoder Architecture

We selected two modern decoder architectures for our study: (i) a conventional convolutional decoder (Choy et al., 2019) frequently employed in 3D semantic segmentation, and (ii) a Transformer decoder that has demonstrated state-of-the-art results in the same field (Fig. 11).

Table 9. Online continual learning in 3D semantic segmentation with Transformer decoder (TD) vs the Convolutional decoder (CD) on ScanNet-val (Dai et al., 2017b) after 3, 5, 10 and 20 (final) tasks.

Method	Mem. Size slots · $ C $	Mem./Data (percent)	mIoU after t tasks			
			3	5	10	20
FT+TD(lower bound)	-	-	1.4	1.6	0.4	0.2
FT+CD(lower bound)	-	-	24.9	3.9	2.1	0.4
MAS+TD (Aljundi et al., 2018)	-	-	1.4	1.6	0.8	0.4
MAS+CD (Aljundi et al., 2018)	-	-	33.7	22.6	10.8	4.9
LwP+TD (Li & Hoiem, 2017)	-	-	34.3	19.3	9.3	4.8
LwP+CD (Li & Hoiem, 2017)	-	-	41.2	34.1	17.4	7.8
ER-50+TD (Chaudhry et al., 2019)	50 · 20	4.1	40.0	35.2	22.9	11.5
ER-50+CD (Chaudhry et al., 2019)	50 · 20	4.1	29.6	30.3	17.4	17.0
ER-200+TD (Chaudhry et al., 2019)	200 · 20	16.4	50.9	36.6	28.6	20.9
ER-200+CD (Chaudhry et al., 2019)	200 · 20	16.4	42.2	36.6	26.8	35.7
ER-400+TD (Chaudhry et al., 2019)	400 · 20	32.8	46.2	40.3	33.0	28.1
ER-400+CD (Chaudhry et al., 2019)	400 · 20	32.8	43.6	36.4	39.0	42.1
LG-Clip+TD-weak sup.	-	-	36.1	20.8	7.7	5.2
LG-Clip+CD-weak sup.	-	-	37.5	18.5	10.4	3.2

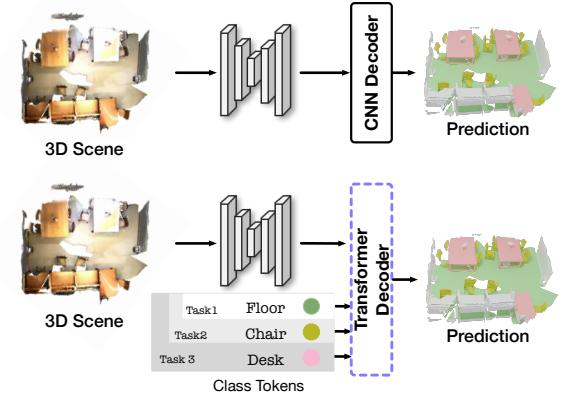


Figure 11. Decoder architectures. We selected two modern decoder architectures for our study: (i) a conventional convolutional decode (top) and a Transformer decoder that has demonstrated state-of-the-art results in the same field (bottom).

Notably, recent efforts (Cermelli et al., 2023; Shang et al., 2023) have explored leveraging the attention mechanism of Transformers to enhance the stability-plasticity trade-off in continual learning. In the Transformer architecture, we learn a set of class tokens but we dynamically add them to the network one task at a time. Those class tokens interact with encoder features in a Transformer decoder, which will predict the class segmentation mask.

However, despite the widespread use of Transformer architectures, we illustrate in Tab. 9 that continual learning challenges persist even when employing such powerful architectures.

The challenges associated with FT are considerably more pronounced in an architecture that demands a substantial amount of data like the Transformer architecture. MAS also encounters significant difficulties in alleviating forgetting in the context of such an architecture. On the other hand, ER exhibits superior performance with the Transformer decoder, however only evident at most up to the 10th task. Surprisingly, ER with a transformer decoder appears to face more substantial challenges compared to the convolutional decoder in later tasks, notably in the 20th task (last column). This observation underscores the importance of evaluating a sufficient number of tasks to make accurate assessments about the effectiveness of different setups in continual learning. Given that the evaluations for 3-5 tasks might not be indicative of performance trends extending to the 20th task and beyond.

D. Implementation Details

Model Encoder. For all datasets and methods, we employ a Res16UNet34A (Choy et al., 2019) as the encoder. The final feature layer has been adjusted to generate features with

Table 10. Hyperparameters for our experiments. LR represents the learning rate and CE the cross-entropy loss. λ_{method} is the weight of the regularization loss depending on the method (*e.g.* λ_{mas} for the MAS loss explained in Eq. 4). We will publish our code to facilitate reproducibility.

Dataset	Methods				
	MAS (Aljundi et al., 2018)	LwF (Li & Hoiem, 2017)	ER (Chaudhry et al., 2019)	LG-Clip	
ScanNet (Dai et al., 2017b)	LR Encoder	10^{-4}	5×10^{-3}	10^{-3}	10^{-3}
	LR Decoder	10^{-5}	10^{-4}	10^{-4}	10^{-4}
	λ_{CE}	1	1	1	-
	λ_{method}	50	1	-	1
S3DIS (Armeni et al., 2016)	LR Encoder	10^{-4}	5×10^{-3}	10^{-3}	10^{-4}
	LR Decoder	10^{-4}	10^{-4}	10^{-4}	10^{-4}
	λ_{CE}	1	1	1	-
	λ_{method}	50	1	-	1
SemanticKITTI (Behley et al., 2019)	LR Encoder	5×10^{-3}	5×10^{-3}	10^{-3}	10^{-4}
	LR Decoder	10^{-4}	10^{-4}	10^{-4}	10^{-4}
	λ_{CE}	1	1	1	-
	λ_{method}	1	1	-	1

a dimensionality of 512, deviating from the original 128 in ME (Choy et al., 2019) to align with the CLIP (Radford et al., 2021) feature dimensionality. Joint Training (JT) was conducted with the 512 dimensionality, showing no significant performance change compared to the original ME (Choy et al., 2019).

Model Decoder. The conventional decoder consists of a $1 \times 1 \times 1$ convolutional layer with stride 1 that projects encoder features to semantic logits. The Transformer decoder consists of 3 layers with 512 dimensions. Each layer consists of self-attention, cross-attention, and feed-forward network. The attention in each layer has 8 heads.

Other Details. For the joint training and (Yang et al., 2023), the model was trained according to the established pipelines of the original methods. For the continual learning setup, the network was trained using cross-entropy loss (except in cases where Dice loss was incorporated), employing a stochastic gradient descent optimizer with a mini-batch size of 10. We train the model with the Transformer decoder using AdamW (Loshchilov & Hutter, 2017). Method parameters in detail:

- **ER:** The mini-batch size was reduced to 5, and the mini-batch size retrieved from the memory buffer was also set to 5, resulting in a total of 10, consistent with other methods.
- **LwF:** We set the temperature factor $T = 2$ as per the original paper and other continual learning (CL) papers.
- **LG-Clip:** Uses only 10 points per scene for supervision.
- **Dice loss:** Weighted with $\lambda_{dice} = 50$ and $\lambda_{CE} = 1$.

For the hyperparameter tuning, we followed (Ghunaim et al., 2023). For details please see Tab. 10. We will publish our code upon acceptance to facilitate the reproducibility of our results.

D.1. Compared Methods

In addition to the Joint Training (JT) and Finetuning (FT) baselines outlined in the main paper, we provide a more detailed description of the seminal continual learning works analyzed in our setup.

Memory Aware Synapses (MAS). MAS (Aljundi et al., 2018) incorporates a penalty to regularize the update of model parameters that are important to past tasks. During each training step, MAS optimizes the following loss:

$$L = \lambda_{CE} \cdot L_{CE}(F(x_k; \theta), \tilde{y}_t) + \lambda_{mas} \cdot \sum_{i,j} \Omega_{ij} (\theta_{ij}^t - \theta_{ij}^{t-1})^2 \quad (4)$$

with θ_{ij}^{t-1} the parameters of the model for the previous task and θ_{ij}^t the current model parameters. It allows the change of parameters that are not important for previous tasks (low Ω_{ij}) and penalizes the change of important ones (high Ω_{ij}). The parameter importance is computed as:

$$\Omega_{ij} = \frac{1}{N} \sum_{k=1}^N \|g_{ij}(x_k)\| \quad (5)$$

where $g_{ij}(x_k) = \frac{\partial [l_2^2(F(x_k; \theta))]}{\partial \theta_{ij}}$ the gradient of the learned function with respect to parameter θ at the data point x_k . N is the total number of points at a given task.

Learning without Forgetting (LwF). LwF (Li & Hoiem, 2017) utilizes knowledge distillation from a teacher to a student model to preserve knowledge from past tasks. The teacher model is the model after learning the last task $t-1$, and the student model is the model to be trained on the current task t . For a new task t with (x_k, y_t) the input x_k and ground truth y_t , LwF computes $F(x_k; \theta^{t-1})$, the output of the previous model θ^{t-1} for the new data x_k . During training, LwF optimizes the following loss:

$$L = \lambda_{CE} \cdot L_{CE}(F(x_k; \theta), \tilde{y}_t) + \\ \lambda_{lwf} \cdot L_{lwf}(F(x_k; \theta^{t-1}), F(x_k; \theta^t)) \quad (6)$$

where $F(x_k; \theta^{t-1})$, $F(x_k; \theta^t)$ are the predicted values of the previous and current model using the same x_k . The λ controls the favoring of the old tasks over the current task.

Experience Replay (ER). ER ([Chaudhry et al., 2019](#)) is a straightforward yet effective replay method. It employs reservoir sampling ([Vitter, 1985](#)) for updating the memory with new examples and random sampling for retrieving examples from memory. Reservoir sampling ensures that each incoming data point has an equal probability of $(\text{memory size})/n$ to be stored in the memory buffer. n is the number of data points observed for each task up to the present. In our configuration, all tasks have the same size memory buffer. ER trains the model by combining in the mini-batch the current task data with the data from memory using the cross-entropy loss.