ReFu: Recursive Fusion for Exemplar-Free 3D Class-Incremental Learning

Yi Yang[1]   Lei Zhong[1]   Huiping Zhuang[2]

[1]The University of Edinburgh   [2]South China University of Technology

{y.yang-249, L.Zhong-10}@sms.ed.ac.uk   hpzhuang@scut.edu.cn

Abstract

We introduce a novel Recursive Fusion model, dubbed ReFu, designed to integrate point clouds and meshes for exemplar-free 3D Class-Incremental Learning, where the model learns new 3D classes while retaining knowledge of previously learned ones. Unlike existing methods that either rely on storing historical data to mitigate forgetting or focus on single data modalities, ReFu eliminates the need for exemplar storage while utilizing the complementary strengths of both point clouds and meshes. To achieve this, we introduce a recursive method which continuously accumulates knowledge by updating the regularized auto-correlation matrix. Furthermore, we propose a fusion module, featuring a Pointcloud-guided Mesh Attention Layer that learns correlations between the two modalities. This mechanism effectively integrates point cloud and mesh features, leading to more robust and stable continual learning. Experiments across various datasets demonstrate that our proposed framework outperforms existing methods in 3D class-incremental learning. Project Page: https://arlo397.github.io/ReFu/

Keywords: Class-Incremental Learning, 3D Computer Vision, Multi-modal Learning.

1. Introduction

Class-incremental learning (CIL) [29, 48, 60, 61] is a machine learning paradigm in which models learn new classes incrementally over time without forgetting previously acquired knowledge. However, CIL faces a significant challenge known as catastrophic forgetting [10], where new data can cause a model to lose information from earlier stages.

Building on the principles of CIL, 3DCIL extends this approach to 3D data, which is increasingly important in applications like robotics [1, 20] and autonomous driving [31, 46, 55]. Recent replay-based methods [4, 7–9, 57] in 3DCIL have made progress, but they rely on storing and replaying subsets of previously encountered data as exemplar—a strategy that can be particularly impractical in scenarios with limited storage capacity [24]. This challenge is
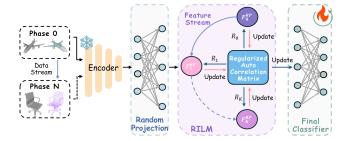

Figure 1. **Overall scheme of our single-modality Framework:** *RePoint* and *ReMesh*, which consist of a frozen encoder, a random projection layer, the RILM memory module, and a final classifier. RILM facilitates continual learning by recursively updating the regularized auto-correlation matrix, ensuring retention of previously learned categories without forgetting. (Sec.3.4, Sec.3.5).

further exacerbated by the complex nature and large size of 3D data, making space limitations even more severe.

Moreover, existing 3DCIL research mainly focuses on single-modality approaches, typically using point clouds [4, 5, 7, 8, 25, 57]. While point clouds effectively capture the overall shape of objects, they lack inherent structural organization. In contrast, mesh data consists of interconnected vertices and faces, forming continuous surfaces that provide more detailed and coherent structural information than point clouds [23]. This disparity raises an important question: *Could we leverage the strengths of mesh modalities to assist in 3D point cloud continual learning?*

In this work, we present Recursive Fusion model, dubbed ReFu, a novel framework for exemplar-free 3D Class-Incremental Learning that integrates both point clouds and meshes. At the core of our approach is the Recursive Incremental Learning Mechanism (RILM), which serves as an memory module for 3D data. Rather than storing data exemplars, RILM updates regularized auto-correlation matrices recursively as new data batches are introduced. By maintaining only the updated matrices, RILM overcomes memory constraints and ensures continuous learning without the need for exemplar storage, providing an effective solution for 3D CIL.

Leveraging RILM, we first develop two single-modality

frameworks: *RePoint* for point clouds and *ReMesh* for meshes, designed to handle incremental learning tasks independently within their respective modalities, as shown in Fig. 1. Our experiments demonstrate that both RePoint and ReMesh achieve state-of-the-art performance in their respective 3D incremental learning scenarios. To further enhance performance, we introduce a modality fusion module with a Pointcloud-guided Mesh Attention Layer that adaptively learns correlations between point cloud and mesh features. This attention mechanism computes a spatial map to align the modalities, enabling efficient feature fusion and capturing more comprehensive 3D representations before processing through RILM.

Experiments on the ModelNet40 [52], SHREC11 (Pointcloud), and their corresponding mesh datasets Manifold [15] and SHREC11 [22] demonstrate that RePoint and ReMesh surpass other baseline models in their respective domains, excelling in both Average Incremental Accuracy and lower Retention Drop across various incremental phases. ReFu, which integrates knowledge from both data modalities, further achieves superior performance. More importantly, unlike previous methods that require storing exemplars [4, 7, 8, 57], ReFu accumulates knowledge by continuously updating regularized auto-correlation matrices without the need for exemplar storage.

To summarize, our contributions are:

1) To our knowledge, ReFu is the first work focus on the multimodal exemplar-free 3DCIL problem using purely 3D data.

2) We introduce a Recursive Incremental Learning Mechanism with a novel fusion module that combines point cloud and mesh features, enabling exemplar-free knowledge accumulation and enhancing 3D representations for robust 3DCIL performance.

3) Experiments demonstrate that ReFu sets a new state of the art in multimodal exemplar-free 3D class-incremental learning, achieving superior results compared to baseline methods.

2. Related Works

2.1. Class-Incremental Learning in 2D:

Class-Incremental Learning (CIL) has garnered considerable attention in recent years, with approaches broadly divided into replay-based [2, 13, 26, 27, 37, 40, 41, 47] and exemplar-free methods [17, 21, 32, 43, 50, 51, 59].

Replay-based methods leverage stored samples or features from previous tasks to reinforce the model's memory while learning new tasks. iCaRL [40] first introduced this approach by selecting samples close to the average embeddings of their respective classes to serve as exemplars. FOSTER [47] further utilizes a small memory buffer, introducing a two-stage learning process that first expands

the model to handle residuals between new and old categories, followed by compression through distillation. Despite their success, replay-based methods are constrained by privacy [6] and memory concerns due to the need to store original samples.

Exemplar-free methods, on the other hand, avoid revisiting historical data and are generally categorized into three main streams: regularization-based, prototype-based, and analytic learning-based methods.

a) Regularization-based methods prevent catastrophic forgetting by constraining changes to previously learned parameters when learning new tasks. Learning without Forgetting (LwF) [21] uses knowledge distillation to maintain consistent outputs across old and new tasks, thereby preserving prior knowledge.

b) Prototype-based methods improve memory retention by leveraging old class prototypes or generating features that integrate past and present knowledge. Recent approaches [32, 50, 51] employ prompt-based mechanisms to encode both shared and task-specific knowledge, while SimpleCIL [59] simplifies the process by using a frozen pretrained model with prototype features of new classes.

c) Analytic learning-based methods are a novel category within continual learning, distinguished by their equivalence between continual learning and joint learning frameworks. ACIL [63] and RanPAC [30] tackle CIL through recursive and iterative processes, while GKEAL [62] leverages a Gaussian kernel process to enhance learning in fewshot scenarios, achieving strong performance in data-scarce environments.

2.2. Class-Incremental Learning in 3D:

Compared to 2D class-incremental learning, research on continual learning in the 3D domain has been relatively limited [4, 5, 7, 8, 25, 44, 53, 57]. I3DOL [7] introduces adaptive geometric structures and geometric-aware attention mechanisms, along with fairness compensation strategies and exemplar support. InOR-Net [8] extends this by incorporating category-guided geometric reasoning and critic-induced geometric attention mechanisms, further improving 3D feature recognition. It also uses a dual adaptive fairness compensation strategy to balance performance between new and old tasks. RCR [57] employs random downsampling and reconstruction loss to compress point cloud data, maintaining memory of prior tasks. [4] utilize Microshapes to extract shared geometric features, reducing cross-task knowledge loss in few-shot 3D learning. These methods all use small memory buffers to preserve previous task knowledge during incremental learning.

In contrast, earlier works like L3DOC [25] and [5] explored continual learning via knowledge distillation. L3DOC utilizes a shared knowledge base and hierarchical point-knowledge factorization for task-specific knowledge

activation, while [5] use soft labels from prior models, combined with semantic information, to help the new model retain old class knowledge.

3. Methods

We start by introducing the necessary preliminaries in Section 3.1. Section 3.2 provides an overview of the proposed model. In Section 3.3, we introduce our fusion module, which combines the point-cloud guided mesh attention and the fusion layer for ReFu. Sections 3.4 and 3.5 detail the random projection layer and the Recursive Incremental Learning Mechanism (RILM), which are designed to ensure stable memory retention and efficient incremental learning.

3.1. Preliminaries

Class-Incremental Learning (CIL) aims to incrementally learn new classes while retaining knowledge from previous phases. In each phase n, the training set for single-modality is denoted as $\mathcal{D}_{s,n}^{\text{train}} = \left\{ \boldsymbol{X}_{s,n}^{\text{train}}, \boldsymbol{Y}_{s,n}^{\text{train}} \right\}$, where $\boldsymbol{X}_{s,n}^{\text{train}}$ represents the input data s (single modality either point cloud or mesh), and $\boldsymbol{Y}_{s,n}^{\text{train}}$ is the corresponding one-hot encoded label set. For multi-modality, the training set is $\mathcal{D}_n^{\text{train}} = \left\{ \boldsymbol{X}_{p,n}^{\text{train}}, \boldsymbol{X}_{m,n}^{\text{train}}, \boldsymbol{Y}_n^{\text{train}} \right\}$, where both point cloud $\boldsymbol{X}_{p,n}^{\text{train}}$ and mesh $\boldsymbol{X}_{m,n}^{\text{train}}$ data share the same label set $\boldsymbol{Y}_n^{\text{train}}$.

During each incremental phase, the model is only allowed to access the current phase's training data. The newly introduced classes are disjoint from previous ones, i.e., $\boldsymbol{Y}_n \cap \boldsymbol{Y}_{n'} = \varnothing$ for $n \neq n'$. After training on the current phase's data, the model is evaluated on the test set consisting of all seen classes up to phase n, represented as $\mathcal{Y}_n^{\text{test}} = \boldsymbol{Y}_1^{\text{test}} \cup \cdots \cup \boldsymbol{Y}_n^{\text{test}}$.

3.2. Model Overview

Our proposed approach comprises two single-modality frameworks, RePoint for point clouds and ReMesh for meshes (Fig. 1), alongside a multimodal framework, ReFu, which integrates both modalities to enhance performance in 3D Class-Incremental Learning (3DCIL) (Fig. 2).

For RePoint and ReMesh, we employ self-supervised pre-trained models, PointMAE [33] and MeshMAE [23], as encoders. These models are chosen for their label- and class-free pre-training [14], making them ideal for incremental learning tasks. Each framework also incorporates a random projection layer for feature expansion and the Recursive Incremental Learning Mechanism (RILM) for efficient knowledge retention.

ReFu integrates both modalities through our fusion module, as discussed in Section 3.3. The rest of the process is similar to RePoint and ReMesh, with frozen encoders helps mitigating catastrophic forgetting [11].

3.3. Fusion

Unlike single-modality methods, our ReFu framework emphasizes effective modality integration by first training the fusion backbone on the ShapeNet dataset [3]. Given a point cloud input $\boldsymbol{X}_p^{\text{train}}$ and a corresponding mesh input $\boldsymbol{X}_m^{\text{train}}$, the point cloud and mesh encoders extract point cloud features $\boldsymbol{F}_p \in \mathbb{R}^{K \times d}$ and mesh features $\boldsymbol{F}_m \in \mathbb{R}^{K \times d}$, respectively, where N represents the number of input samples and d is the dimensionality of the feature space.

To effectively combine these multimodal features, we employ the Pointcloud-guided Mesh Attention Layer to adaptively learn the correlations between the point cloud and mesh features. We first compute the spatial attention map through the following steps:

$$
\begin{aligned}
\boldsymbol{Score}_p &= tanh\left(\boldsymbol{F}_p \boldsymbol{W}_p\right) \\
\boldsymbol{Score}_m &= tanh\left(\boldsymbol{F}_m \boldsymbol{W}_m\right) \\
\boldsymbol{W}_{\text{Spa}} &= Softmax\left(\boldsymbol{Score}_m \circ \boldsymbol{Score}_p\right)
\end{aligned}
\tag{1}
$$

where $\boldsymbol{W}_p \in \mathbb{R}^{d \times d}$ and $\boldsymbol{W}_m \in \mathbb{R}^{d \times d}$ are learnable projection matrices. The function $tanh(\cdot)$ serves as the non-linear activation function, and $\circ$ denotes the Hadamard product operation. The resulting spatial attention map $\boldsymbol{W}_{\text{Spa}} \in \mathbb{R}^{d \times d}$ is used to weight the mesh features, producing the weighted mesh features:

$$
\boldsymbol{F}_m' = \boldsymbol{W}_{\text{Spa}} \boldsymbol{F}_m
\tag{2}
$$

where $\boldsymbol{F}_m'$ represents the mesh features after spatial attention weighting.

Finally, the processed point cloud and mesh features are concatenated and passed to a classifier to obtain the final prediction:

$$
\hat{y} = \text{CLS}\left(concat\left(\boldsymbol{F}_p, \boldsymbol{F}_m'\right)\right),
\tag{3}
$$

where $concat\left(\boldsymbol{F}_p, \boldsymbol{F}_m'\right)$ denotes the concatenated feature representation, $\text{CLS}(\cdot)$ represents the classifier. The output $\hat{y}$ is the predicted class label.

3.4. RILM Alignment

For the initial phase 0 in incremental learning, the data $\boldsymbol{X}_{s,0}^{\text{train}}$ is passed through the encoders to extract the feature matrix. This matrix is then processed by a two-layer linear feed-forward network, where the first layer performs feature expansion via random projection, and the second layer is used for classification.

Specifically, we feed the input samples $\boldsymbol{X}_{s,0}^{\text{train}}$ of the classes in phase 0 into the encoder to extract the embeddings for point clouds or meshes, which are then expanded using random projection and processed by an activation function:

$$
\boldsymbol{F}_0^{\text{RP}} = \boldsymbol{\sigma}\left(Encoder\left(\boldsymbol{X}_{s,0}^{\text{train}}\right) \boldsymbol{W}^{\text{RP}}\right)
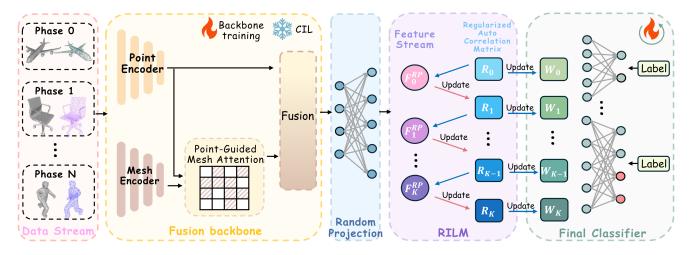\tag{4}
$$

3

Figure 2. **Overview of our ReFu framework.** During incremental learning, data flows progressively into the model. As outlined in Section 3.3, our proposed fusion backbone, pre-trained on the ShapeNet dataset and frozen during learning, extracts and fuses features from point clouds and meshes. These fused features are then expanded via a random projection layer and input into the Recursive Incremental Learning Mechanism (RILM). RILM recursively updates the regularized auto-correlation matrix and classifier weights. Only the matrix and weights from the previous phase $(n-1)$ are stored, without retaining any raw data as exemplar.

Here, $\boldsymbol{\sigma}$ is the activation function. $\boldsymbol{W}^{\mathrm{RP}}$ denotes the parameters of the random projection layer, and $\boldsymbol{F}_0^{\mathrm{RP}} \in \mathbb{R}^{N_0 \times d_{\mathrm{RP}}}$, where d_{RP} is the expanded dimensionality. The random projection layer expands the feature space, introducing additional parameters and improving memory retention in RILM, leading to enhanced model performance, as analyzed in Section 4.7.

Next, we map the expanded embeddings to the label matrix $\boldsymbol{Y}_0^{\mathrm{train}}$ via a linear classifier layer, whose weights are computed by solving the following:

$$\underset{\boldsymbol{W}_0}{\mathrm{argmin}} = \left\| \boldsymbol{Y}_0^{\mathrm{train}} - \boldsymbol{F}_0^{\mathrm{RP}} \boldsymbol{W}_0 \right\|_F^2 + \eta \left\| \boldsymbol{W}_0 \right\|_F^2 \quad (5)$$

where $\| \cdot \|_F$ denotes the Frobenius norm, and η is the regularization term. The optimal solution is then obtained as:

$$\hat{\boldsymbol{W}}_0 = \left(\left(\boldsymbol{F}_0^{\mathrm{RP}} \right)^\top \boldsymbol{F}_0^{\mathrm{RP}} + \eta \boldsymbol{I} \right)^{-1} \left(\boldsymbol{F}_0^{\mathrm{RP}} \right)^\top \boldsymbol{Y}_0^{\mathrm{train}} \quad (6)$$

where $\hat{\boldsymbol{W}}_0$ represents the estimated parameters of the linear classifier layer, and $^\top$ denotes the transpose operation.

3.5. Class-Incremental Learning

Following the RILM alignment of point cloud and mesh embeddings, we proceed to the class-incremental learning (CIL) steps. Specifically, the learning problem using all seen data up to phase $n-1$ can be extended from (5) as:

$$\underset{\boldsymbol{W}_{n-1}}{\mathrm{argmin}} \left\| \boldsymbol{Y}_{0:n-1}^{\mathrm{train}} - \boldsymbol{F}_{0:n-1}^{\mathrm{RP}} \boldsymbol{W}_{n-1} \right\|_F^2 + \eta \left\| \boldsymbol{W}_{n-1} \right\|_F^2 \quad (7)$$

where

$$\boldsymbol{Y}_{0:n-1}^{\mathrm{train}} = \begin{bmatrix} \boldsymbol{Y}_0^{\mathrm{train}} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{Y}_{n-1}^{\mathrm{train}} \end{bmatrix}, \quad \boldsymbol{F}_{0:n-1}^{\mathrm{RP}} = \begin{bmatrix} \boldsymbol{F}_0^{\mathrm{RP}} \\ \vdots \\ \boldsymbol{F}_{n-1}^{\mathrm{RP}} \end{bmatrix} \quad (8)$$

Similar to (6), the solution to this optimization problem can be obtained as:

$$\hat{\boldsymbol{W}}_{n-1} = \left(\sum_{n'=0}^{n-1} \boldsymbol{A}_{n'} + \eta \boldsymbol{I} \right)^{-1} \left(\sum_{n'=0}^{n-1} \boldsymbol{C}_{n'} \right) \quad (9)$$

where

$$\begin{cases} \boldsymbol{A}_{n'} = \left(\boldsymbol{F}_{n'}^{\mathrm{RP}} \right)^\top \boldsymbol{F}_{n'}^{\mathrm{RP}} \\ \boldsymbol{C}_{n'} = \left(\boldsymbol{F}_{n'}^{\mathrm{RP}} \right)^\top \boldsymbol{Y}_{n'}^{\mathrm{train}} \end{cases} \quad (10)$$

Here, $\boldsymbol{A}_{n'}$ represents the auto-correlation feature matrix, and $\boldsymbol{C}_{n'}$ represents the cross-correlation matrix. To further simplify the equation, let:

$$\boldsymbol{R}_{n-1} = \left(\sum_{n'=0}^{n-1} \boldsymbol{A}_{n'} + \eta \boldsymbol{I} \right)^{-1} \quad (11)$$

to be the regularized auto-correlation matrix. We can redefine the CIL process as a recursive least squares task as outlined in the following theorem.

Theorem 1. Given training data $\mathcal{D}_n^{\mathrm{train}}$ and the estimated weights of the final classifier layer $\hat{\boldsymbol{W}}_{n-1}$ from phase $n-1$, the updated weights $\hat{\boldsymbol{W}}_n$ can be recursively obtained by:

$$\hat{\boldsymbol{W}}_n = \hat{\boldsymbol{W}}_{n-1} - \boldsymbol{R}_n \boldsymbol{A}_n \hat{\boldsymbol{W}}_{n-1} + \boldsymbol{R}_n \boldsymbol{C}_n \quad (12)$$

4

with

$$\boldsymbol{R}_n = \boldsymbol{R}_{n-1} - \boldsymbol{R}_{n-1} \left(\boldsymbol{F}_n^{\mathrm{RP}} \right)^{\top}$$
$$\times \left(\boldsymbol{F}_n^{\mathrm{RP}} \boldsymbol{R}_{n-1} \left(\boldsymbol{F}_n^{\mathrm{RP}} \right)^{\top} + \boldsymbol{I} \right)^{-1} \boldsymbol{F}_n^{\mathrm{RP}} \boldsymbol{R}_{n-1} \quad (13)$$

Proof. See Supplementary Material for details.

Summary: Theorem 1 proves that joint training can be transformed into a recursive incremental learning process. Instead of requiring access to the entire dataset at once, we only need to retain the weights $\hat{\boldsymbol{W}}_{n-1}$ and the regularized auto-correlation matrix $\boldsymbol{R}_{n-1}$ from the previous phase. By recursively updating these with new data, while keeping the fusion backbone frozen, the class-incremental learning (CIL) process becomes mathematically equivalent to joint training. This demonstrates the strong memory retention capability of our RILM.

4. Experiments

4.1. Datasets

We employ the ModelNet40 [52] dataset as our primary source for rigid object classification and the SHREC11 [22] dataset to evaluate performance on non-rigid 3D objects. ModelNet40, which has similar structure to the ShapeNet [3] dataset used for pretraining, contains 12,311 3D models, with 9,843 used for training and 2,468 for testing, spanning 40 distinct categories. Point cloud representations were generated by uniformly sampling 8,192 points from each object. For mesh data, due to the presence of holes or boundaries on the surface of the original ModelNet40 data, we follow the data pre-processing protocol of Mesh-MAE [23] and SubdivNet [15], generating remeshed watertight or 2-manifold meshes, referred to as the Manifold40 dataset. No data augmentation techniques were applied to either the ModelNet40 or Manifold40 datasets.

In contrast, the SHREC11 dataset comprises 30 classes with 20 samples per class, focusing primarily on non-rigid 3D objects. These objects exhibit greater morphological flexibility, with their shapes undergoing significant changes under different poses and motions. This dataset was selected to evaluate the robustness of our method when applied to out-of-domain data. For point clouds, we uniformly sampled 8,192 points per object, while for meshes, we adopted the pre-processing protocol from SubdivNet [15].

4.2. Class-Incremental Setting

We evaluate class-incremental learning under two settings: a short-range setup with 10 phases and a long-range setup with N phases, where N represents the total number of classes. For ModelNet40 (Manifold40), $N = 40$, and for SHREC11, $N = 30$. In the short-range setup, each phase introduces 4 classes for ModelNet40 and 3 classes for SHREC11, while the long-range setup adds 1 class per phase for both datasets.

4.3. Evaluation Metrics

Evaluation metrics are designed to assess two critical aspects: Incremental Learning Capability and Knowledge Retention. To evaluate these dimensions, we adopt metrics similar to those used in [56], focusing on Average Incremental Accuracy ($\mathcal{A}$) and Retention Drop ($\mathcal{R}$).

For assessing Incremental Learning Capability, we employ the Average Incremental Accuracy ($\mathcal{A}$) metric. $\mathcal{A}$ provides a comprehensive measure of a model's performance across all learning phases and is calculated as $\mathcal{A} = \frac{1}{N} \sum_{n=1}^{N} \mathcal{A}_n$, where $\mathcal{A}_n$ represents the average test accuracy at phase n across all previously seen classes. A higher $\mathcal{A}$ value indicates the model's effectiveness in maintaining high performance as it incrementally learns new classes.

To evaluate Knowledge Retention, we introduce the Retention Drop ($\mathcal{R}$) metric. $\mathcal{R}$ quantifies performance degradation as the model learns new tasks and is defined as $\mathcal{R} = \mathcal{A}_1 - \mathcal{A}_N$, where $\mathcal{A}_1$ is the test accuracy after the initial phase, and $\mathcal{A}_N$ is the accuracy after the final phase N. This metric measures the model's ability to retain knowledge throughout incremental learning. A lower $\mathcal{R}$ value indicates better retention, reflecting less degradation on early learned tasks as new tasks are introduced.

4.4. Implementation details

We conduct all experiments using PyTorch [35]. Following the approach in [4,44], we replace the encoder in 2DCIL methods with pre-trained PointMAE and MeshMAE models, using them as feature extractors for point cloud and mesh datasets, respectively. For the training of ReFu's fusion backbone, we freeze the self-supervised pre-trained point cloud and mesh encoders and fine-tune the remaining components of the model (pointcloud-guided mesh attention layer, fusion layer, and classifier). We use the Adam optimizer [16] with a learning rate of 4e-3 and a batch size of 24. After this training phase, the parameters of the backbone are fixed and align with those of RePoint and ReMesh, serving as feature extractors in the subsequent incremental learning steps.

4.5. Performance of *RePoint* and *ReMesh*

To assess the efficacy of the newly developed RePoint and ReMesh frameworks, we conducted comparative analyses against various incremental learning methodologies: 1) Fine-tuning, where the model is initialized with the parameters from the previous phase and re-trained on new data batches; 2) LwF [21], which utilizes the preservation of outputs from previous examples as a regularizer for new tasks;

Table 1. **Performance comparison on point cloud datasets**, evaluated using $\mathcal{A}$ and $\mathcal{R}$. Bold values denote the overall best results, while underlined values highlight the top-performing baselines. Red-highlighted rows indicate our RePoint method, and blue-highlighted rows represent ReFu. ReFu is separated from other methods as it leverages both point cloud and mesh inputs, provided here only for direct comparison (see Section 4.6).

| Method | Exemplar-free? | SHREC11 (Point cloud) | | | | ModelNet40 | | | |
| | | 10 Phase | | 30 Phase | | 10 Phase | | 40 Phase | |
		$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓
Fine-tuning	✓	30.39	87.34	13.32	96.67	28.26	89.62	9.70	99.18
LwF [21]	✓	42.21	81.27	12.32	94.52	48.80	75.52	11.59	95.94
SimpleCIL [59]	✓	76.56	28.37	77.11	31.15	79.77	25.65	81.28	33.06
iCaRL [40]	×	<u>89.56</u>	17.02	81.73	27.70	87.01	27.22	82.18	36.71
Foster [47]	×	88.56	<u>14.92</u>	<u>92.36</u>	<u>14.33</u>	<u>92.74</u>	<u>14.33</u>	<u>92.70</u>	<u>14.38</u>
RePoint (Ours)	✓	**94.82**	**8.73**	**94.61**	**9.29**	**96.51**	**7.65**	**96.85**	**8.20**
ReFu (Ours)	✓	**96.40**	**6.73**	**96.36**	**6.93**	**97.42**	**5.12**	**97.21**	**6.51**

Table 2. **Performance comparison on mesh datasets**, evaluated using $\mathcal{A}$ and $\mathcal{R}$. Bold values denote the overall best results, while underlined values highlight the top-performing baselines. Red-highlighted rows indicate our ReMesh method, and blue-highlighted rows represent ReFu. ReFu is separated from other methods as it leverages both point cloud and mesh inputs, provided here only for direct comparison (see Section 4.6).

| Method | Exemplar-free? | SHREC11 (Mesh) | | | | Manifold40 | | | |
| | | 10 Phase | | 30 Phase | | 10 Phase | | 40 Phase | |
		$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓
Fine-tuning	✓	28.65	90.00	13.31	96.67	26.86	88.61	9.70	99.19
LwF [21]	✓	34.68	86.67	13.31	96.67	36.13	87.57	14.35	95.95
SimpleCIL [59]	✓	86.29	18.77	86.87	18.77	77.48	27.91	79.26	36.06
iCaRL [40]	×	85.20	21.42	84.22	25.79	76.14	38.03	73.79	47.11
Foster [47]	×	<u>92.94</u>	<u>13.73</u>	<u>91.83</u>	<u>13.41</u>	<u>86.10</u>	<u>22.81</u>	<u>83.66</u>	<u>25.28</u>
ReMesh (Ours)	✓	**95.19**	**7.82**	**95.54**	**8.21**	**94.08**	**11.31**	**94.51**	**12.61**
ReFu (Ours)	✓	**96.40**	**6.73**	**96.36**	**6.93**	**97.42**	**5.12**	**97.21**	**6.51**

3) iCaRL [40], a method that combines exemplars with distillation to prevent forgetting; 4) SimpleCIL [59], using a fixed feature extractor from a pretrained model and prototypes of each class as classifier weights; 5) Foster [47], which enhances the model's adaptability to new categories by integrating new trainable feature extractors with the existing model and maintains a portion of old data to preserve the memory of previous categories.

We compare our proposed RePoint and ReMesh methods with the baseline approaches in Tables 1 and 2. The results show that both RePoint and ReMesh consistently outperform the compared methods, demonstrating strong performance in 3D class-incremental learning.

For RePoint, on the SHREC11 dataset, our method increases $\mathcal{A}$ by 5.26% and decreases $\mathcal{R}$ by 6.19% in the 10-phase setting. In the 30-phase setting, it improves $\mathcal{A}$ by 2.25% and reduces $\mathcal{R}$ by 5.04%. On ModelNet40, RePoint surpasses Foster by 3.77% in $\mathcal{A}$ and reduces $\mathcal{R}$ by 6.68% in the 10-phase scenario, with gains of 4.15% in $\mathcal{A}$ and a reduction of 6.18% in $\mathcal{R}$ in the 40-phase setting.

For ReMesh, on SHREC11, it improves $\mathcal{A}$ by 2.25% and decreases $\mathcal{R}$ by 5.91% in the 10-phase setting, and by 3.71% in $\mathcal{A}$ and 5.2% in $\mathcal{R}$ in the 30-phase setting. On Manifold40, ReMesh outperforms Foster by 7.98% in $\mathcal{A}$ and decreases $\mathcal{R}$ by 11.5% in the 10-phase scenario, with gains of 10.85% in $\mathcal{A}$ and reductions of 12.67% in $\mathcal{R}$ in the 40-phase setting.

Furthermore, we present the testing accuracy at each incremental step for RePoint, ReMesh, and other baselines in Fig. 3. Our methods consistently achieve top performance
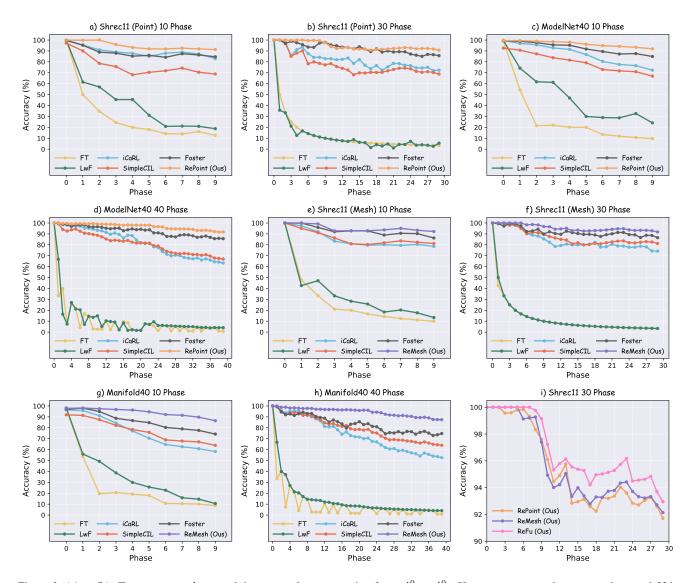
Figure 3. **(a) ~ (h):** Test accuracy $\mathcal{A}_n$ at each incremental stage, ranging from $\mathcal{A}_1^0$ to $\mathcal{A}_N^0$. Here, n represents the current phase and N is the total number of phases. The results are based on the PointMAE / MeshMAE backbone. We report accuracy on point cloud datasets: (1) SHREC11 (Point) and (2) ModelNet40, and mesh datasets: (3) SHREC11 and (4) Manifold40, under 10 phase and N phase settings. **(i)** shows the test accuracy of our proposed three frameworks, ReFu, RePoint and Remesh.

in most steps in nearly all incremental steps and outperform all the baselines in the final phases. Similar to the exemplar-free SimpleCIL, RePoint and ReMesh also freeze the backbone during incremental learning, but use a more effective memory mechanism instead of relying on prototypes. In summary, our methods demonstrate clear advantages in 3DCIL, validating their effectiveness.

4.6. Performance of *ReFu*

The fusion of 3D modalities in ReFu further enhances recognition accuracy and knowledge retention in CIL tasks. In Tables 1 and 2, we also provide the performance of ReFu for comparison with RePoint and ReMesh. The results in-

dicate that ReFu outperforms the single-modality methods, demonstrating the effectiveness of multimodal fusion. Furthermore, in Fig. 3 (i), we present the testing accuracy at each incremental step for ReFu, RePoint, and ReMesh on the SHREC11 30-phase task. It can be observed that ReFu consistently achieves better performance than the single-modality methods at all stages.

4.7. Ablation Studies

Random Projection: As presented in Table 3, Random Projection (RP) is a key component of our model, with its dimensionality governed by the parameter $d_{(\mathrm{rp})}$. By increasing the feature dimension, RP effectively captures ad-

Table 3. Performance of ReFu methods over 10 phases on SHREC11 and ModelNet40 (Manifold40) datasets, evaluated with $\mathcal{A}$ and $\mathcal{R}$. Red-highlighted rows indicate results with Random Projection (RP) applied.

Dataset	With RP?	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓
SHREC11	×	94.13	10.82
	✓	96.40 (+2.27)	6.73 (-4.09)
ModelNet40 (Manifold40)	×	94.45	12.01
	✓	97.42 (+2.97)	5.12 (-6.89)

Table 4. Performance of ReFu methods during 10 phases. Metrics used are $\mathcal{A}$ and $\mathcal{R}$. For clarity, fusion methods are abbreviated as follows: AddF = addition, ConcatF = concatenation, AttnF = attention-guided concatenation.

Dataset	Fusion Methods	$\mathcal{A}$ (%) ↑	$\mathcal{R}$ (%) ↓
SHREC11	AddF	95.23	8.27
	ConcatF	96.32 (+1.09)	7.25 (-1.02)
	AttnF	96.40 (+0.08)	6.73 (-0.52)
ModelNet40 (Manifold40)	AddF	95.55	7.74
	ConcatF	97.14 (+1.59)	6.13 (-1.61)
	AttnF	97.42 (+0.28)	5.12 (-1.01)

ditional information and alleviates the over-fitting issues typically encountered in recursive learning approaches [63]. To evaluate its impact, we performed a 10-phase incremental learning experiment using ReFu on the SHREC11 and ModelNet40 datasets, where the feature dimensionality was expanded by 12 times compared to the original.

Fusion Methods: We first assess the effectiveness of our fusion strategy by comparing it to a variant utilizing simple addition fusion. As shown in Table 4, addition fusion yields sub-optimal performance. On the SHREC11 dataset, the average accuracy ($\mathcal{A}$) is 95.23%, only marginally out-performing the single-modality ReMesh baseline (95.19%). On ModelNet40, $\mathcal{A}$ drops below that of RePoint (96.51%), indicating that addition fusion struggles to leverage complementary information between modalities. When one modality under-performs, the overall result is adversely impacted, limiting the potential benefits of multimodal fusion.

In contrast, concatenation fusion delivers superior results. On SHREC11, it improves $\mathcal{A}$ by 1.09% over addition fusion and reduces retention drop ($\mathcal{R}$) by 1.02%. On ModelNet40, it further increases $\mathcal{A}$ by 1.59% and reduces $\mathcal{R}$ by 1.61%. These findings suggest that concatenation fusion preserves more information from both modalities, mitigating knowledge forgetting in continual learning.

Nevertheless, simple concatenation does not truly "fuse" the features from different modalities. To address this, we employ attention-guided concatenation fusion. On SHREC11, attention-guided concatenation boosts $\mathcal{A}$ by an additional 0.08% over standard concatenation and reduces $\mathcal{R}$ by 0.52%. On ModelNet40, it further raises $\mathcal{A}$ by 0.28% and decreases $\mathcal{R}$ by 1.01%. These results demonstrate that attention-guided fusion effectively enhances cross-modal interaction, leading to improved performance in 3D Class-Incremental Learning (3DCIL).

5. Discussion

Why we use 2D CIL methods as baselines? Currently, there are relatively few 3DCIL approaches [9] specifically designed for meshes, with most research focused on point

clouds. For example, methods like RCR [57] have show that point cloud data, even after significant compression, can still serve as effective exemplars. However, there is no clear evidence suggesting that such techniques can be applied with equal effectiveness to mesh data. In addition, given the extensive research in 2DCIL [12, 42, 45, 54], most 3DCIL approaches [5, 7, 8, 25, 44] adopt 2D strategies by replacing 2D encoders with 3D counterparts, thereby facilitating the transfer of 2D CIL techniques to 3D tasks.

The Role of Large Pretrained Models in 3DCIL: Current 3DCIL methods heavily rely on early feature extractors like PointNet [38], PointNet++ [39], and DGCNN [49]. However, leveraging more powerful backbone networks is crucial for improving the performance of 3D continual learning. Studies like [44] show that using PointCLIP [58] can further improve results. In 2DCIL, SimpleCIL [59] demonstrated that frozen pretrained models (PTMs) can even outperform advanced CIL methods by simply updating the classifier with prototype features. PTMs, trained on large datasets, transfer knowledge efficiently and generalize well, making pretrained models a new paradigm in continual learning [18, 19, 28, 34, 36]. Motivated by this, we introduce the MAE [14] pretraining paradigm into 3DCIL, enhancing knowledge transfer and performance.

6. Conclusion

In this paper, we introduce the Recursive Fusion model (ReFu), a novel framework for exemplar-free 3D Class-Incremental Learning that integrates both point cloud and mesh modalities. Our model employs the Recursive Incremental Learning Mechanism (RILM) to accumulate knowledge without storing exemplars, recursively updating regularized auto-correlation matrices. We develop two specialized models: *RePoint* for point clouds and *ReMesh* for meshes, both achieving state-of-the-art performance in 3D incremental learning tasks. ReFu further enhances 3D representation learning with the Pointcloud-guided Mesh Attention Layer. Extensive experiments on multiple datasets validate the effectiveness of our approach.

References

[1] Ali Ayub and Carter Fendley. Few-shot continual active learning by a robot. *NIPS*, 2022.

[2] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, 2021. 2

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 3, 5

[4] Townim Chowdhury, Ali Cheraghian, Sameera Ramasinghe, Sahar Ahmadi, Morteza Saberi, and Shafin Rahman. Few-shot class-incremental learning for 3d point cloud objects. In *ECCV*, 2022. 1, 2, 5

[5] Townim Chowdhury, Mahira Jalisha, Ali Cheraghian, and Shafin Rahman. Learning without forgetting for 3d point cloud objects. In *Advances in Computational Intelligence: 16th International Work-Conference on Artificial Neural Networks, IWANN*, 2021. 1, 2, 3, 8

[6] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *PAMI*, 2021. 2

[7] Jiahua Dong, Yang Cong, Gan Sun, Bingtao Ma, and Lichen Wang. I3dol: Incremental 3d object learning without catastrophic forgetting. In *AAAI*, 2021. 1, 2, 8

[8] Jiahua Dong, Yang Cong, Gan Sun, Lixu Wang, Lingjuan Lyu, Jun Li, and Ender Konukoglu. Inor-net: Incremental 3-d object recognition network for point cloud representation. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 1, 2, 8

[9] Tom Fischer, Yaoyao Liu, Artur Jesslen, Noor Ahmed, Prakhar Kaushik, Angtian Wang, Alan Yuille, Adam Kortylewski, and Eddy Ilg. inemo: Incremental neural mesh models for robust class-incremental learning. *ECCV*, 2024. 1, 8

[10] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 1999. 1

[11] Yulu Gan, Yan Bai, Yihang Lou, Xianzheng Ma, Renrui Zhang, Nian Shi, and Lin Luo. Decorate the newcomers: Visual domain prompt for continual test time adaptation. In *AAAI*, 2023. 3

[12] Zhanxin Gao, Jun Cen, and Xiaobin Chang. Consistent prompting for rehearsal-free continual learning. In *CVPR*, 2024. 8

[13] Saisubramaniam Gopalakrishnan, Pranshu Ranjan Singh, Haytham Fayek, Savitha Ramasamy, and Arulmurugan Ambikapathi. Knowledge capture and replay for continual learning. In *WACV*, 2022. 2

[14] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 3, 8

[15] Shi-Min Hu, Zheng-Ning Liu, Meng-Hao Guo, Jun-Xiong Cai, Jiahui Huang, Tai-Jiang Mu, and Ralph R Martin. Subdivision-based mesh convolution networks. *TOG*, 2022. 2, 5

[16] Diederik P Kingma. Adam: A method for stochastic optimization. *ICLR*, 2015. 5

[17] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 2017. 2

[18] Jaewoo Lee, Jaehong Yoon, Wonjae Kim, Yunji Kim, and Sung Ju Hwang. Stella: Continual audio-video pre-training with spatiotemporal localized alignment. In *ICML*, 2024. 8

[19] Kuan-Ying Lee, Yuanyi Zhong, and Yu-Xiong Wang. Do pre-trained models benefit equally in continual learning? In *WACV*, 2023. 8

[20] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information fusion*, 2020. 1

[21] Zhizhong Li and Derek Hoiem. Learning without forgetting. *PAMI*, 2017. 2, 5, 6

[22] Z Lian, A Godil, B Bustos, M Daoudi, J Hermans, S Kawamura, Y Kurita, G Lavoua, P Dp Suetens, et al. Shape retrieval on non-rigid 3d watertight meshes. In *Eurographics workshop on 3d object retrieval*, 2011. 2, 5

[23] Yaqian Liang, Shanshan Zhao, Baosheng Yu, Jing Zhang, and Fazhi He. Meshmae: Masked autoencoders for 3d mesh data analysis. In *ECCV*, 2022. 1, 3, 5

[24] Bo Liu, Xuesu Xiao, and Peter Stone. A lifelong learning approach to mobile robot navigation. *IEEE Robotics and Automation Letters*, 2021. 1

[25] Yuyang Liu, Yang Cong, Gan Sun, Tao Zhang, Jiahua Dong, and Hongsen Liu. L3doc: Lifelong 3d object classification. *TIP*, 2021. 1, 2, 8

[26] Yaoyao Liu, Yingying Li, Bernt Schiele, and Qianru Sun. Online hyperparameter optimization for class-incremental learning. In *AAAI*, 2023. 2

[27] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *CVPR*, 2021. 2

[28] Zuxin Liu, Jesse Zhang, Kavosh Asadi, Yao Liu, Ding Zhao, Shoham Sabach, and Rasool Fakoor. Tail: Task-specific adapters for imitation learning with large pretrained models. *ICLR*, 2024. 8

[29] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. Class-incremental learning: survey and performance evaluation on image classification. *PAMI*, 2022. 1

[30] Mark D McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. Ranpac: Random projections and pre-trained models for continual learning. *NIPS*, 2024. 2

[31] M Jehanzeb Mirza, Marc Masana, Horst Possegger, and Horst Bischof. An efficient domain-incremental learning approach to drive in all weather conditions. In *CVPR*, 2022. 1

[32] Jun-Yeong Moon, Keon-Hee Park, Jung Uk Kim, and Gyeong-Moon Park. Online class incremental learning on stochastic blurry task boundary via mask and visual prompt tuning. In *ICCV*, 2023. 2

[33] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *ECCV*, 2022. 3

[34] Keon-Hee Park, Kyungwoo Song, and Gyeong-Moon Park. Pre-trained vision and language transformers are few-shot incremental learners. In *CVPR*, 2024. 8

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NIPS*, 2019. 5

[36] Weiguo Pian, Shentong Mo, Yunhui Guo, and Yapeng Tian. Audio-visual class-incremental learning. In *ICCV*, 2023. 8

[37] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *ECCV*, 2020. 2

[38] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 8

[39] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NIPS*, 2017. 8

[40] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 2, 6

[41] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *NIPS*, 2019. 2

[42] Anurag Roy, Riddhiman Moulick, Vinay K Verma, Saptarshi Ghosh, and Abir Das. Convolutional prompting meets language models for continual learning. In *CVPR*, 2024. 8

[43] Filip Szatkowski, Mateusz Pyla, Marcin Przewięźlikowski, Sebastian Cygert, Bartłomiej Twardowski, and Tomasz Trzciński. Adapt your teacher: Improving knowledge distillation for exemplar-free continual learning. In *WACV*, 2024. 2

[44] Yuwen Tan and Xiang Xiang. Cross-domain few-shot incremental learning for point-cloud recognition. In *WACV*, 2024. 2, 5, 8

[45] Minh-Tuan Tran, Trung Le, Xuan-May Le, Mehrtash Harandi, and Dinh Phung. Text-enhanced data-free approach for federated class-incremental learning. In *CVPR*, 2024. 8

[46] Thanh-Dat Truong, Pierce Helton, Ahmed Moustafa, Jackson David Cothren, and Khoa Luu. Conda: Continual unsupervised domain adaptation learning in visual perception for self-driving cars. In *CVPR*, 2024. 1

[47] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. In *ECCV*, 2022. 2, 6

[48] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: theory, method and application. *PAMI*, 2024. 1

[49] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 2019. 8

[50] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *ECCV*, 2022. 2

[51] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *CVPR*, 2022. 2

[52] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 2, 5

[53] Yuwei Yang, Munawar Hayat, Zhao Jin, Chao Ren, and Yinjie Lei. Geometry and uncertainty-aware 3d point cloud class-incremental semantic segmentation. In *CVPR*, 2023. 2

[54] Yiwen Ye, Yutong Xie, Jianpeng Zhang, Ziyang Chen, Qi Wu, and Yong Xia. Continual self-supervised learning: Towards universal multi-modal medical data representation learning. In *CVPR*, 2024. 8

[55] Liangqi Yuan, Yunsheng Ma, Lu Su, and Ziran Wang. Peer-to-peer federated continual learning for naturalistic driving action recognition. In *CVPR*, 2023. 1

[56] Xianghu Yue, Xueyi Zhang, Yiming Chen, Chengwei Zhang, Mingrui Lao, Huiping Zhuang, Xinyuan Qian, and Haizhou Li. Mmal: Multi-modal analytic learning for exemplar-free audio-visual class incremental tasks. In *ACMMM*, 2024. 5

[57] Maciej Zamorski, Michał Stypułkowski, Konrad Karanowski, Tomasz Trzciński, and Maciej Zięba. Continual learning on 3d point clouds with random compressed rehearsal. *Computer Vision and Image Understanding*, 2023. 1, 2, 8

[58] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. In *CVPR*, 2022. 8

[59] Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pretrained models: Generalizability and adaptivity are all you need. *IJCV*, 2024. 2, 6, 8

[60] Da-Wei Zhou, Hai-Long Sun, Jingyi Ning, Han-Jia Ye, and De-Chuan Zhan. Continual learning with pre-trained models: A survey. In *IJCAI*, 2024. 1

[61] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Class-incremental learning: A survey. *PAMI*, 2024. 1

[62] Huiping Zhuang, Zhenyu Weng, Run He, Zhiping Lin, and Ziqian Zeng. Gkeal: Gaussian kernel embedded analytic learning for few-shot class incremental task. In *CVPR*, 2023. 2

[63] Huiping Zhuang, Zhenyu Weng, Hongxin Wei, Renchunzi Xie, Kar-Ann Toh, and Zhiping Lin. Acil: Analytic class-incremental learning with absolute memorization and privacy protection. *NIPS*, 2022. 2, 8