L3DOC: Lifelong 3D Object Classification

Yuyang Liu[1,2,3],   Yang Cong[1,2],   Gan Sun[1,2]

[1]State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China
[2]Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang, China
[3]University of Chinese Academy of Sciences, China

{liuyuyang, congyang}@sia.cn   sungan1412@gmail.com

Abstract

3D object classification has been widely-applied into both academic and industrial scenarios. However, most state-of-the-art algorithms are facing with a fixed 3D object classification task set, which cannot well tackle the new coming data with incremental tasks as human ourselves. Meanwhile, the performance of most state-of-the-art lifelong learning models can be deteriorated easily on previously learned classification tasks, due to the existing of unordered, large-scale, and irregular 3D geometry data. To address this challenge, in this paper, we propose a Lifelong 3D Object Classification (i.e., L3DOC) framewor, which can consecutively learn new 3D object classification tasks via imitating "human learning". Specifically, the core idea of our proposed L3DOC model is to factorize PointNet in a perspective of lifelong learning, while capturing and storing the shared point-knowledge in a perspective of layer-wise tensor factorization architecture. To further transfer the task-specific knowledge from previous tasks to the new coming classification task, a memory attention mechanism is proposed to connect the current task with relevant previously tasks, which can effectively prevent catastrophic forgetting via soft-transferring previous knowledge. To our best knowledge, this is the first work about using lifelong learning to handle 3D object classification task without model fine-tuning or retraining. Furthermore, our L3DOC model can also be extended to other backbone network (e.g., PointNet++). To the end, comparisons on several point cloud datasets validate that our L3DOC model can reduce averaged $1.68 \sim 3.36\times$ parameters for the overall model, without sacrificing classification accuracy of each task.

1 Introduction

With the development of 3D perception technology, a variety of applications in 3D object recognition are developed, *e.g.,* automatic driving [2], robotics [37], bioanalysis [16], medical diagnostic [8] and environmental perception [34, 39], etc. Furthermore, 3D object recognition is a classification-based recognition problem. Some 3D classification algorithms
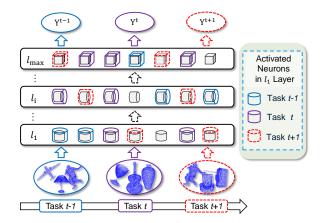

Figure 1: Illustration of lifelong 3D object classification problem, where different shapes and colors denote different layers and object classification tasks, respectively, and dotted line denotes the new $(t+1)$-th classification task.

have achieved remarkably progressions, *e.g.,* the well-known PointNet [30] learns to encode each point to a global point cloud signature and PointNet++ [31] focuses on capturing local structure in point space.

However, as shown in Figure 1, most current 3D object classification models are developed for classification tasks with fixed distribution, which can result in hugely time-consuming and tedious when handling new tasks via simply retraining. Additionally, a series of new 3D classification tasks in real-world applications may have the following settings: the sequence of tasks has no definite rule, tasks are unpredictably switched, and any individual task may not be repeated frequently due to multiple environmental factors (*e.g.,* illumination, occlusion and clutter). Merely fine-tuning or retraining the current model for new tasks can ignore the knowledge learned from previously learned tasks, which can cause performance degradation due to catastrophic forgetting. Therefore, this paper explores how to continually learn a series of 3D object classification tasks, while achieving less memory and higher efficiency.

A pioneering solution to these above problems is to employ lifelong machine learning methods. Generally, the fundamental goal of lifelong learning is to achieve better performance or faster convergence/training-speed for a new task, based on leveraging the knowledge of previously tasks [44]. Although lifelong learning has been successfully applied into reinforcement learning [25, 46] and metric learning [38], most state-of-the-arts are more concerned with processing 2D data [19, 36]. To solve the 3D object classification problem from the perspective of lifelong learning, two main challenges need to be considered:

- **Knowledge inside 3D Geometry Data:** The unordered and point-independent properties of point cloud data will lead to some existing knowledge transfer mechanisms [28, 19, 17, 13, 45, 18, 33] that cannot capture the shared knowledge from the geometry data according to common sense.

- **Catastrophic Forgetting:** Different point cloud objects can have similar approximation or geometric data among continual classification tasks, *e.g.,* tables and desks in Modelnet40 [42] dataset. Therefore, it is reasonable to overwrite previous task-specific knowledge to ensure the excellent adaptability of new tasks.

To overcome these challenges above, we develop a lifelong 3D object classification (L3DOC) framework, which intends to capture and store the shared point-knowledge from previously tasks, and adapt to new tasks without degrading performance of the previously ones. More specifically, a layer-wise tensor-channel factorization architecture is proposed to capture the unique shared point-knowledge, which can preserve the neighborhood structure of the unordered point cloud data for each task. A memory attention mechanism is employed to recognize the unknown differences between tasks, and selectively soft-transfer task-specific knowledge to learning new classification tasks. Therefore, our proposed L3DOC model can learn new point cloud object environments end-to-end with more compact parameters, while retaining the performance of the past ones. Finally, we evaluate our model by factorizing the well-known PointNet [30] network and use it on three public point cloud benchmarks for 3D object classification, and ours have faster efficiency and excellent robustness without causing catastrophic forgetting. Meanwhile, our framework can be easily extended to other backbone networks, *e.g.,* PointNet++ [31].

The contribution of this paper is threefold:

- We propose a lifelong 3D object classification (L3DOC) framework, which can transfer previous point-knowledge to learn new classification task without sacrifice classification accuracy. To our best knowledge, this is the first work about 3D object classification from the perspective of lifelong learning.

- We develop a general shared point-knowledge factorization architecture at the channels of convolution kernel, which can accommodate the point-knowledge among

previously learned tasks and preserve unordered neighborhood distribution in point cloud.

- To further reduce catastrophic forgetting the knowledge of previously tasks, we present a memory attention mechanism to selectively pick out the task-specific knowledge and soft-transfer it to learn the new tasks. Various experiment show the efficiency and effectiveness of our proposed L3DOC model.

2 Related Work

In this section, we review related work on 3D object classification and recognition and lifelong learning, as well as works related to preventing the catastrophic forgetting problems.

2.1 3D Object Classification and Recognition

Generally, there are three common strategies for 3D object classification and recognition: **Feature-Based Strategy** [15, 24], which usually extracts distinctive invariant (*i.e.,* image scale and rotation invariant) local features from 2D color image and back project them to 3D space; **Template-Based Strategy** [1, 3, 6, 21], which extracts template features from the scanning model under multi-view, and searches the optimal matching by sliding windows, *e.g.,* LineMod [12] achieve robust 3D object classification and recognition by embedding quantized image contours and standard orientations on RGBD images; **Learned-Based Strategy** [41, 10, 11], various deep learning methods have shown their superior performance in 3D object classification and recognition, *e.g.,* PointNet [30] and PointNet++ [31] directly process raw unordered point cloud data and achieve better performance by addressing the invariance to permutations and transformations of the inputs.

Although these methods above show high performance, their real-world applications inevitably rely on a large number of invariantly distributed training data [21]. Therefore, some solutions in specific applications based on multi-task learning are proposed, *e.g.,* 3D Bi-Ventricular Segmentation of Cardiac Images task [8], 3D Action Recognition [20, 27], and 3D Semantic-Instance Segmentation [29]. However, most multi-task learning focus on maximizing the performance of all the tasks by sharing knowledge, which consumes a lot of memory and computation [36].

2.2 Lifelong Learning

Lifelong learning [40, 36] is often regarded as online multi-task learning, which focuses more on efficiently learning continuous tasks based on previous knowledge while optimizing the performance across all tasks seen so far. To share fixed network parameters, they [4, 35, 32] are widely used but are more limited by known tasks. As a promotion, some methods [9, 5, 23] for automatically selecting parameters and network sizes have been proposed. As more complex task relationships are learned, some frameworks [14, 22, 44] design

dynamic network extensions for task-specific parts (*e.g.*, filters, and neurons), which only considers the input and output spaces in general neural network architecture and ignore the critical characteristic of task-specific parts in space and channel. For the computational efficiency, *i.e.*, ELLA [36] assumes network parameters θ_t for task t is a linear combination of latent shared knowledge L and task-specific sparse tensors s_t (*i.e.*, $\theta_t = Ls_t$) [36]. However, this linear combination cannot accommodate the knowledge embedded in deep neural network. Fortunately, the deconvolutional factorized CNN (DF-CNN) [19] algorithm for 2D neat data proposes a deconvolution mapping and a tensor contraction to factorize the convolution kernel: $F_t^{(l)} = \text{deconv}(\widehat{L}^{(l)}; V_t^{(l)}) \bullet U_t^{(l)}$, where $\widehat{L}^{(l)} \in \mathbb{R}^{l_h \times l_w \times l_c}$ is the latent knowledge base in the l-th convolutional layer of each task-specific CNN_t, $V_t^{(l)} \in \mathbb{R}^{v_h \times v_w \times u \times l_c}$ is the deconvolution kernel of $\widehat{L}^{(l)}$, and $U_t^{(l)} \in \mathbb{R}^{u \times f_{in} \times f_{out}}$ is the partial factor of $F_t^{(l)}$. When DF-CNN facing with 3D point cloud data, it inevitably causes the inefficiency since the dimension of $U_t^{(l)}, V_t^{(l)}, \widehat{L}^{(l)}$ is larger than corresponding 3D convolutional kernel. Therefore, in order to address the massive challenge for 3D geometry data, we take the first attempt to achieve lifelong 3D object classification by storing the shared point-knowledge and selectively transferring the previous task-specific knowledge.

3 L3DOC: Lifelong 3D Object Classification

In this section, we comprehensively introduce our proposed lifelong 3D object classification framework. We first state the lifelong 3D object classification problem, followed by our proposed L3DOC framework.

3.1 3D Lifelong Object Classification Problem

Given a set of m 3D object classification tasks $\mathcal{T}^1, \ldots, \mathcal{T}^m$, where each individual task $\mathcal{T}^t$ consumes a set of objects $X^t \in \mathbb{R}^{o_t} = \{x_1^t, x_2^t, ..., x_{o_t}^t\}$ and true label $Y^t \in \mathbb{R}^{o_t \times c_t}$ as inputs, where $x_i^t \in \mathbb{R}^{n \times d} (\forall i = 1, \ldots, o_t)$ is an unordered point set, o_t is the number of object for task t, c_t is the number of class for the t-th task, n and d are the number of each point set and the dimension of each point, respectively. The corresponding output for the t-th task is estimated label $\widetilde{Y^t} \in \mathbb{R}^{o_t \times c_t}$ for all the c_t candidate classes. Formally, an associated neural network PointNet [30] (this work focuses on the classification setting, which can be extended into semantic segmentation setting) for task t can be formulated as:

$$f(X^t; \theta_t) = \rho(\text{MAX}_{i=1,..,n}\{h(x_i^t)\}), \tag{1}$$

where θ_t denotes the network parameters of task $\mathcal{T}^t$, $h(\cdot)$ is approximated by a multi-layer perceptron network, and $\rho(\cdot)$ is a continuous function.

However, continually learning 3D object classification tasks without storing the network parameters of previously learned tasks is not considered by the pioneering works. In the lifelong 3D object classification setting, the system faces with a set of 3D classification tasks $\mathcal{T}^1, \ldots, \mathcal{T}^m$ without the prior knowledge (*e.g.*, task order, distribution, etc), where each $\mathcal{T}^t$ is defined in Eq. (1). The target of this system is to establish a lifelong 3D object classification framework such that: **1)** the system can ensure the classification accuracy of new tasks $\mathcal{T}^t$ without catastrophic forgetting the performance of old ones $\mathcal{T}^1...\mathcal{T}^{t-1}$; **2)** in the learning period of each new task, the convergence speed of lifelong learning system can be faster than traditional 3D object classification model, without sacrificing large memory.

3.2 The Proposed Method

Different from 2D neat data, each point in 3D object classification task is processed independently and identically, which makes it necessary to use the convolution kernel $W_t^{(l)} \in \mathbb{R}^{1 \times 1 \times w_{\text{in}} \times w_{\text{out}}}$ in the convolution process, *i.e.*, the convolution kernel in the multi-layer perceptron of PointNet [30]. To achieve lifelong 3D object classification task, as shown in Figure 2, we first introduce how to capture the knowledge in 3D geometry data via a layer-wise point-knowledge factorization architecture, and then present a memory attention mechanism to further prevent catastrophic forgetting.

3.2.1 Layer-Wise Point-Knowledge Factorization

Limited by the special distribution of neighborhood of 3D point cloud data, the size of convolution kernel is set as $(1 \times 1 \times w_{\text{in}} \times w_{\text{out}})$ in PointNet [30], so it is necessary to decompose the convolution kernel's input and output channels in a tensor manner. Formally, each convolution kernel tensor $W_t^{(l)} \in \mathbb{R}^{1 \times 1 \times w_{\text{in}} \times w_{\text{out}}}$ in the l-th layer of the task $\mathcal{T}^t$ can be decomposed into two factors:

$$W_t^{(l)} = \sum_{k=1}^{n} C_{t,(1,1,k)}^{(l)} D_{t,(k,w_{\text{in}},w_{\text{out}})}^{(l)} = C_t^{(l)} D_t^{(l)}, \tag{2}$$

where $C_t^{(l)} \in \mathbb{R}^{1 \times 1 \times n}$ is a task-specific factor and $D_t^{(l)} \in \mathbb{R}^{n \times w_{\text{in}} \times w_{\text{out}}}$ is a task-specific intermediate factor of shared point-knowledge. Intuitively, this decomposition operator allows knowledge to be shared across different tasks, where parameter n can decide how important the knowledge is to current task $\mathcal{T}^t$, and be controlled by the convolution kernel output size:

$$n = w_{\text{out}}/\hat{n}, \tag{3}$$

where $\hat{n}$ is the shrinkage scale for $n \ll (w_{\text{in}} \times w_{\text{out}})$, and can be used as adjustments for retaining task-specific information. Then the intermediate factor $D_t^{(l)}$ can be obtained from the shared point-knowledge $L^{(l)}$ by deconvolution mapping:
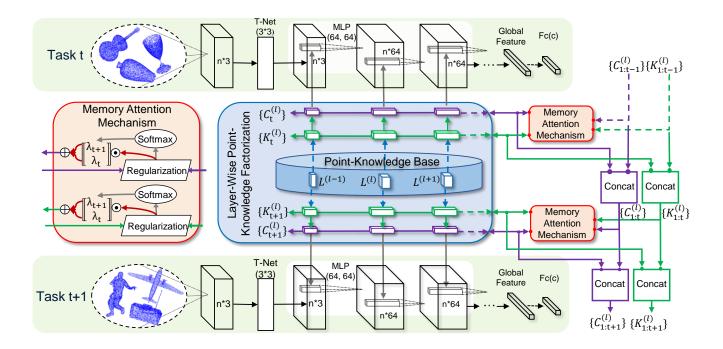
Figure 2: The illustration of our L3DOC framework, which shows the transfer and update of the shared point-knowledge of all the layers in $\mathcal{T}^t, \mathcal{T}^{t+1}$. The convolution kernel $W^{(l)}$ (grey boxes) is reconstructed from the shared point-knowledge base $L^{(l)}$ (blue boxes) by learning deconvolution mapping ($K^{(l)}$ in green boxes) and tensor contraction ($C^{(l)}$ in purple boxes). Task-specific factors $K_{t+1}^{(l)}$ and $C_{t+1}^{(l)}$ in the new task is inherited from the previously tasks through a memory attention mechanism.

$$D_t^{(l)} = \mathcal{D}_t(L^{(l)}; K_t^{(l)}), \qquad (4)$$

where $L^{(l)} \in \mathbb{R}^{n \times w_{\text{in}} \times l_{\text{out}}}$, $K_t^{(l)} \in \mathbb{R}^{s \times s \times w_{\text{out}} \times l_{\text{out}}}$ is the kernel of task-specific deconvolution mapping $\mathcal{D}_t(\cdot)$, and s is the spatial size of $K_t^{(l)}$. Similarly, the volume of $L^{(l)}$ can be set according to current convolution kernel's size:

$$l_{\text{out}} = w_{\text{out}}/\hat{l}_{\text{out}}, \qquad (5)$$

where $l \ll w_{\text{out}}$. Since adjusting the size of $\hat{l}_{\text{out}}$ can change the storage capacity of point-knowledge between different classification tasks, the parameters of these variables can directly affect the complexity of overall network. Notice that the task-specific tensors $\{\{K_t^{(l)}, C_t^{(l)}\}_{l=1}^{l_{\max}}\}_{t=1}^{t_{\max}}$ only represent a small fraction of all the parameters, the majority of which are contained within the shared point-knowledge base $\{L^{(l)}\}_{l=1}^{l_{\max}}$. Therefore, after expressing new convolution kernels $W_t^{(l)}$ as task-specific tensor factors $K_t^{(l)}, C_t^{(l)}$, one can learn them on $\mathcal{T}^t$ end-to-end.

3.2.2 Factor Regularization

Assuming we can correct various tensor parameters of one task $\mathcal{T}^t$ based on the classification loss $\mathcal{L}_c = \left\| \widetilde{Y}^t - Y^t \right\|_2^2$ in the above model, such a simple reliance on the classification loss $\mathcal{L}_c$ of a single task will cause the model parameters to be overcorrected and cause task-specific of shared point-knowledge base $\{L^{(l)}\}_{l=1}^{l_{max}}$ to be overwritten. Considering

that $\{L^{(l)}\}_{l=1}^{l_{\max}}$ will be retrained when each task arrives, we only use the previously knowledge $\{L_{t-1}^{(l)}\}_{l=1}^{l_{\max}}$ under all the old tasks $\mathcal{T}^1, ..., \mathcal{T}^{t-1}$ to constrain $\{L^{(l)}\}_{l=1}^{l_{\max}}$ for new task $\mathcal{T}^t$. A simple way to prevent the learned $\{L_{t-1}^{(l)}\}_{l=1}^{l_{\max}}$ from being destroyed by $\mathcal{T}^t$ is to impose a ℓ_2-norm regularizer to reduce the excessive knowledge gap between different tasks among all the layers:

$$\mathcal{L}_L = \sum_{l=1}^{l_{\max}} \left\| L_{t-1}^{(l)} - L^{(l)} \right\|_2^2. \qquad (6)$$

With the constraint above, $\{L^{(l)}\}_{l=1}^{l_{\max}}$ will replay previously knowledge $\{L_{t-1}^{(l)}\}_{l=1}^{l_{\max}}$ at all the times without forgetting the performance in $\mathcal{T}^1, ..., \mathcal{T}^{t-1}$ when the task t is updated. Additionally, the performance on new task t should not be lost in our L3DOC model, we thus add ℓ_2-regularization constraints on the key task-specific tensor factors $\{\{K_i^{(l)}, C_i^{(l)}\}_{l=1}^{l_{\max}}\}_{t=1}^{t_{\max}}$ for the task $\mathcal{T}^t$:

$$\mathcal{L}_{K_i} = \sum_{l=1}^{l_{\max}} \left\| K_i^{(l)} - K_t^{(l)} \right\|_2^2,$$
$$\mathcal{L}_{C_i} = \sum_{l=1}^{l_{\max}} \left\| C_i^{(l)} - C_t^{(l)} \right\|_2^2, \qquad (7)$$

where $\mathcal{L}_{K_i}, \mathcal{L}_{C_i}$ denote difference losses between the current task $\mathcal{T}^t$ and each past task $\mathcal{T}^i, \forall i = 1, ..., t-1$.

4

Algorithm 1 Memory Attention Mechanism (MAM)

Input: $\{L_{t-1}^{(l)}, K_{1:t}^{(l)}, C_{1:t}^{(l)}\}_{l=1}^{l_{\max}}, \lambda \geq 0;$
Output: $\mathcal{L}_r$
1: $\mathcal{L}_L \leftarrow$ Regularization($\{L_{t-1}^{(l)}, L^{(l)}\}_{l=1}^{l_{\max}}$) via Eq. (6);
2: **for** $i = 1, ..., t-1$ **do**
3:   $\mathcal{L}_{K_i} \leftarrow$ Regularization($K_i^{(l)}, K_t^{(l)}$) via Eq. (7);
4:   $\mathcal{L}_{C_i} \leftarrow$ Regularization($C_i^{(l)}, C_t^{(l)}$) via Eq. (7);
5:   $(\lambda_{K_i}, \lambda_{C_i}) \leftarrow$ AttentionScore($\mathcal{L}_{K_i}, \mathcal{L}_{C_i}$) via Eq. (8);
6:   $\mathcal{L}_r \leftarrow \lambda \mathcal{L}_L + \sum_{i=1}^{t-1} \lambda_{K_i} \mathcal{L}_{K_i} + \sum_{i=1}^{t-1} \lambda_{C_i} \mathcal{L}_{C_i};$
7: **end for**

3.2.3 Memory Attention Mechanism (MAM)

In the manner described above, structural features of similar point cloud between successive tasks are stored in the shared point-knowledge base $\{L^{(l)}\}_{l=1}^{l_{\max}}$. To selectively transfer this knowledge and further prevent the learned $\{K_i^{(l)}, C_i^{(l)}\}_{l=1}^{l_{\max}}$ from losing their uniqueness, *i.e.*, catastrophic forgetting, we add a memory attention mechanism to each $\{\mathcal{L}_{K_i}, \mathcal{L}_{C_i}\}_{i=1}^{t-1}$. The attention scores $\lambda_{K_i}, \lambda_{C_i}$ for each $\{\mathcal{L}_{K_i}, \mathcal{L}_{C_i}\}_{i=1}^{t-1}$ are set by soft-attention mechanism [43], and can be defined as follows:

$$\lambda_{K_i} = \frac{1}{l_{\max}} \mathcal{H}(\mathcal{L}_{K_i} | \{\mathcal{L}_{K_i}\}_{i=1}^{t-1}),$$
$$\lambda_{C_i} = \frac{1}{l_{\max}} \mathcal{H}(\mathcal{L}_{C_i} | \{\mathcal{L}_{C_i}\}_{i=1}^{t-1}), \quad (8)$$

where $\mathcal{H}(\cdot)$ is the soft-attention mechanism obtaining the attention distribution probability information of each difference loss between $\mathcal{T}^t$ and each $\{\mathcal{T}^i\}_{i=1}^{t-1}$. The attention scores $\{\lambda_{K_i}, \lambda_{C_i}\}_{i=1}^{t-1}$ are the average attention distribution probability among all the layers, which determine how important each past tasks is to current task. Therefore, we can obtain the total loss function $\mathcal{L}_{\text{total}}$ minimized in model:

$$\mathcal{L}_{\text{total}} = \min \mathcal{L}_c + \lambda \mathcal{L}_L + \sum_{i=1}^{t-1} \lambda_{K_i} \mathcal{L}_{K_i} + \sum_{i=1}^{t-1} \lambda_{C_i} \mathcal{L}_{C_i}, \quad (9)$$

where λ controls the dependency of the shared knowledge base on past tasks. Specifically, when the difference of task-special tensors between $\mathcal{T}^i$ and $\mathcal{T}^t$ is large, the value of λ_{K_i} and λ_{C_i} increase, so that the attention of model optimization is paid on $\mathcal{T}^i$; when the difference of special tensors between $\mathcal{T}^i$ and $\mathcal{T}^t$ is small, the value of λ_{K_i} and λ_{C_i} decrease. Therefore, $\mathcal{T}^i$ cannot be considered in model optimization, and the uniqueness of $\{K_i^{(l)}, C_i^{(l)}\}_{l=1}^{l_{max}}$ within a certain range is maintained. In short, this memory attention mechanism can selectively screens out a small number of valuable task-specific tensor differences, and soft-transfer the similar knowledge among previous tasks. Finally, the goal of selectively transferring shared point-knowledge and task-specific knowledge for new task can be achieved. Furthermore, the memory attention mechanism is summarized in **Algorithm 1**, and our L3DOC framework is summarized in **Algorithm 2**.

3.2.4 Complexity Analysis

Meanwhile, the shared point-knowledge in $\{L^{(l)}\}_{l=1}^{l_{\max}}$ can be efficiently transferred to new task by retraining very few parameters while preserving the performance of previously tasks. The convolution kernels $\{W_t^{(l)} \in \mathbb{R}^{(1 \times 1 \times w_{\text{in}} \times w_{\text{out}})}\}_{l=1}^{l_{max}}$ of the multi-layer perceptron (*i.e.*, MLP) in PointNet [30] have $N_W = (3 \times 64 + 64^2 + 64^2 + 64 \times 128 + 128 \times 1024) = 159936$ parameters, and the number of additional parameters in this network are N_{addition}. For all the trained tasks, the total number of parameters needed in STL-PointNet is:

$$N_{\text{STL-PointNet}} = (N_W + N_{\text{addition}}) \times t_{\max}$$
$$\approx N_W \times t_{\max} = 159936 \times t_{\max}, \quad (10)$$

where $N_{\text{addition}} \ll N_W$. The total number of parameters in DF-CNN [19] based on PointNet network is:

$$N_{\text{DF-PointNet}} = (N_U + N_V) \times t_{\max} + N_{\widehat{L}}$$
$$= \sum_{l=1}^{l_{max}} (u \times f_{in}^{(l)} \times f_{out}^{(l)}) \times t_{\max}$$
$$+ (v_n \times v_w \times u \times l_h) \times t_{\max} \quad (11)$$
$$+ l_h \times l_w \times l_c$$
$$= u \times (N_W + v_n \times v_w \times l_h) \times t_{\max}$$
$$+ l_h \times l_w \times l_c$$
$$> N_W \times t_{\max} \approx N_{\text{STL-PointNet}},$$

where $f_{in} = w_{in}$, $f_{out} = w_{out}$, and $u \geq 1$ respect to the setup in DF-CNN, which not only does not reduce the number of parameters in PointNet network, but increases the amount of additional calculations. From this perspective, the direct use of DF-CNN is not applicable in PointNet. In our L3DOC framework , the convolution kernel $\{W_t^{(l)}\}_{l=1}^{l_{max}}$ can be decomposed into smaller task-specific tensors $\{K_t^{(l)}, C_t^{(l)}\}_{l=1}^{l_{\max}}$ in $\mathcal{T}^t$, and shared point-knowledge $\{L^{(l)}\}_{l=1}^{l_{\max}}$, benefitting from the Layer-Wise Point-Knowledge Factorization. The overall parameters can be computed as:

$$N_{\text{OURS}} = (N_C + N_K) \times t_{\max} + N_L$$
$$= \sum_{l=1}^{l_{\max}} \{(n^{(l)} + s^2 \times w_{\text{out}}^{(l)} \times l_{\text{out}}^{(l)}) \times t_{\max}$$
$$+ n^{(l)} \times w_{\text{in}}^{(l)} \times l_{\text{out}}^{(l)}\}$$
$$= \sum_{l=1}^{l_{\max}} \{(\frac{1}{\hat{n}} + \frac{s^2 \times w_{\text{out}}^{(l)}}{\hat{l}_{\text{out}}}) \times w_{\text{out}}^{(l)} \times t_{\max} \quad (12)$$
$$+ \frac{w_{\text{in}}^{(l)} \times w_{\text{out}}^{(l)}{}^2}{\hat{n} \times \hat{l}_{\text{out}}}\}.$$

From Section 3.2.1, it is known that the size of $\hat{n}, \hat{l}_{out}, s$ can affect performance of network, we thus select two better versions of parameter setting, through experiment comparison:

Algorithm 2 Lifelong 3D Object Classification Framework

Input: $\{X^t, Y^t\}_{t=1}^{t_{\max}}$, $\hat{n}$, $\hat{l}_{\text{out}}$, Spatial Size s, $\lambda \geq 0$, MaxEpoch, BacthSize;
Output: $\{\widetilde{Y}^t\}_{t=1}^{t_{\max}}$;
1: $\{L^{(l)}\}_{l=1}^{l_{\max}} \leftarrow$ Initialization$(\hat{n}, \hat{l}_{\text{out}})$;
2: **if** isAnotherClassificationTaskAvailable() **then**
3:   **if** $t = 1$ **then**
4:     $\{K_t^{(l)}, C_t^{(l)}\}_{l=1}^{l_{\max}} \leftarrow$ Initialization$(\hat{n}, \hat{l}_{\text{out}}, s)$;
5:   **else**
6:     $\{K_t^{(l)}, C_t^{(l)}\}_{l=1}^{l_{\max}} \leftarrow \{K_{t-1}^{(l)}, C_{t-1}^{(l)}\}_{l=1}^{l_{\max}}$; $t = t + 1$;
7:   **end if**
8:   BatchIndex $\leftarrow Y^t/$BatchSize;
9:   **while** $i = 1, \ldots, (\text{MaxEpoch} \times \text{BatchIndex})$ **do**
10:     **for** $l = 1, \ldots, l_{\max}$ **do**
11:       $D_t^{(l)} \leftarrow \mathcal{D}_t(L^{(l)}; K_t^{(l)})$ via Eq. (4);
12:       $W_t^{(l)} \leftarrow (C_t^{(l)} \bullet D_t^{(l)})$ via Eq. (2);
13:     **end for**
14:     $\{X_b^t, Y_b^t\} \leftarrow$ getBatchData(BatchSize, $X^t, \widetilde{Y}^t$);
15:     $\widetilde{Y}_b^t \leftarrow$ BackboneNetwork$(W_t^{(l)}, X_b^t, Y_b^t)$;
16:     $\mathcal{L}_c \leftarrow$ getClassifationLoss$(\widetilde{Y}_b^t, Y_b^t)$;
17:     **if** $t = 1$ **then**
18:       $\{L^l, K_t^{(l)}, C_t^{(l)}\}_{l=1}^{l_{\max}} \leftarrow \arg\min_{L,K,C}(\mathcal{L}_c)$;
19:     **else**
20:       $\{L^l, K_t^{(l)}, C_t^{(l)}\}_{l=1}^{l_{\max}} \leftarrow \arg\min_{L,K,C}(\mathcal{L}_c +$ MAM$(\{L_{t-1}^{(l)}, K_{1:t}^{(l)}, C_{1:t}^{(l)}\}_{l=1}^{l_{\max}}, \lambda)$);
21:     **end if**
22:   **end while**
23: **end if**

L3DOC (*Group1*) ($\hat{n}$=16, $\hat{l}_{\text{out}}$=32, s=2), L3DOC (*Group2*): ($\hat{n}$=32, $\hat{l}_{\text{out}}$=32, s=2). Supposing $t_{\max} = 10$, $N_{\text{OURS}} = 950664 \approx N_{\text{STL-PointNet}}$ / 1.68 by following L3DOC (*Group1*), and N_{OURS}=475332 $\approx N_{\text{STL-PointNet}}/3.36$ by following L3DOC (*Group2*). Therefore, our L3DOC framework reduce 1.68~3.36$\times$ parameters less than the STL-PointNet under well performance, which is more compact and more suitable for 3D object classification tasks.

4 Experiments

In this section, we carry out empirical comparisons with the state-of-the art lifelong learning models, followed by the ablation analysis about our L3DOC model.

4.1 Experimental Setup

This subsection provides the detail about benchmarks, comparison models, metrics and implementation details.

Benchmarks: To evaluate 3D object classification tasks, we use the point cloud benchmarks dataset, *i.e.,* ModelNet [42] (ModelNet10 and ModelNet40) and 3DMNIST [7]. Specifically, we use the Modelnet10 and Modelnet40 dataset

processed in PointNet [30] with 1024 points for each object. Corresponding to the setting of lifelong learning, we split ten 3D object classification tasks for the ModelNet10 dataset, where each task has five object categories containing training and testing data; similarly, for the ModelNet40 dataset, we randomly split it into 20 tasks, where each task 10 object categories (*i.e., $c = c_t$* is set as 10). For 3D point cloud handwritten digits (3DMNIST), we sampled its point cloud data into (1024, 3) (*i.e., $n = 1024$ and $d = 3$*) by Furthest-Point-Sampling [26] and split them into ten tasks with five object categories in each task.

Comparison Models: To accurately evaluate our proposed model, we use the PointNet [30] as backbone network and compare our model with the following models: **1)** *STL-PointNet*: Each task is trained independently based on the baseline PointNet [30] network; **2)** *Prog-PointNet*: This model can restore the fixed PointNet [30] network weights for each task during subsequent testing, following the progressive network's [35] setup; **3)** *DF-PointNet*: The key Deconvolutional Factorized solution in Deconvolutional Factorized CNN (DF-CNN) [19] is partially applied to the convolution kernel of PointNet, *i.e.,* MLP. Such a setting can also compare the effectiveness of models more fairly. **4)** *Dynamically Expandable Network (DEN)* [44]: This model can perform the point cloud object recognition task in an end-to-end way, which can dynamically expand the network. **5)** *L3DOC*: From Section 3.2.1, it is known that the size of $\hat{n}, \hat{l}_{\text{out}}, s$ can affect performance of network, we thus select two sets of better parameters through experiment comparison: *L3DOC (Group1):* ($\hat{n} = 16$, $\hat{l}_{\text{out}} = 32$, $s = 2$), *L3DOC (Group2):* ($\hat{n} = 32$, $\hat{l}_{\text{out}} = 32$, $s = 2$).

Metrics: The following metrics is used to demonstrate the performance on sequence tasks during the testing phase.

- *Peak Per-Task Accuracy (PPA)*: The Top 5% test accuracy of each task in its training phase. This metric highlights the peak performance for the current task.
- *Training Time (TT)*: The time required for completing all task training.
- *Average Per-Task Accuracy (APA)[19]*: The average test accuracy of all previously tasks under the current one after it has been trained, *i.e.,* the overall performance of the current task model on all the tasks.
- *Catastrophic Forgetting Ratio (CFR)*: By following [19], this metric denotes the ratio of the test accuracy of all seen tasks to their peak per-task accuracy.
- *Speed of Convergence (SC)[19]*: The convergence speed of training on each task, *i.e.,* the number of required rounds when the test accuracy reaches 98% of the peak accuracy of each task.

Implementation details. We train all models on ModelNet10 and 3DMNIST for 100 epochs, and on ModelNet40 for 160 epochs. The PointNet-based model follows its basic hyper-parameter settings. We perform the point cloud object
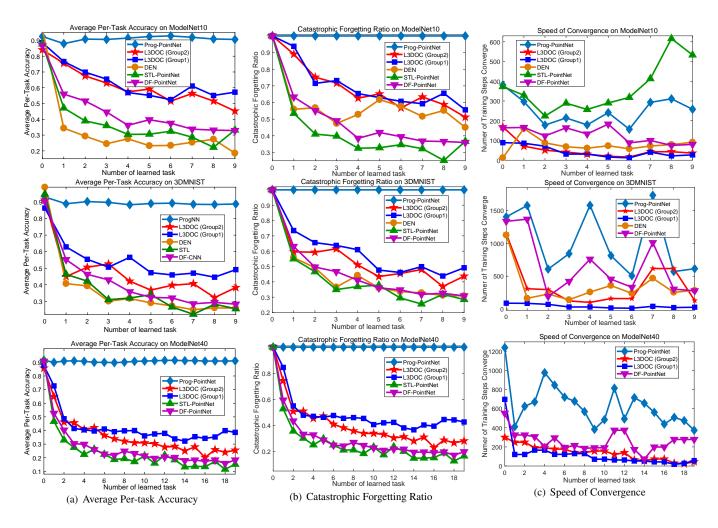
Figure 3: The result of 3D object classification tasks in terms of (a) Average per-task accuracy, (b) Catatrophic forgetting ratio, and (c) Speed of convergence in three point cloud datasets (ModelNet10 (Top), 3DMNIST (Middle) and ModelNet40 (Bottom)). Lines with different colors represent different comparison models. Notice that all the metrics are decreased when increasing the number of training task.

classification task end-to-end in DEN provided by the original author. Additionally, notice that the tasks in the ModelNet40 dataset are significant differences, we do not test DEN model on the Modelnet40 dataset since its network size is excessively increased as a new task comes.

4.2   Experimental Results

This subsection presents experimental results in terms of efficiency and accuracy on three public point cloud benchmarks.

4.2.1   Efficiency Evaluation

This subsection measures the efficiency *v.s. PPA* of each model on all the three datasets. As the results shown in Table 1, notice that **1)** all the PointNet-based models (*e.g.,* our proposed L3DOC, Prog-PointNet and DF-PointNet) can achieve excellent performance in terms of *PPA*, when comparing with STL-PointNet. However, both Prog-PointNet and

DF-PointNet are not faster than that of STL-PointNet, it is because they have a more high-complexity network structure. **2)** All the two versions of our model can effectively perform all the tasks while maintaining strong classification ability, which justifies the rationality of the designed layer-wise point-knowledge factorization operator.

Table 1: Comparison the *Peak Per-Task Accuracy* and *Training Time* (10k second) for all the models on three benchmarks. Models with the best performance are indicated in red.

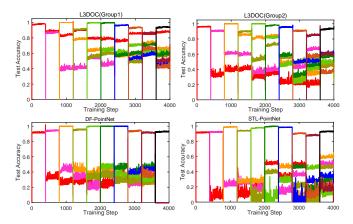| # Benchmarks	ModelNet10		ModeNet40		3DMNIST	
# Model	Peak-Acc	Time	Peak-Acc	Time	Peak-Acc	Time
Prog-PointNet [35]	100.00	3.21	100.00	8.09	100.00	3.66
STL-PointNet [30]	98.48	1.30	99.39	2.84	99.41	1.32
DEN [44]	53.22	0.42	-	-	81.11	0.87
DF-PointNet [19]	100.00	3.11	100.00	3.85	100.00	1.62
L3DOC(Group1)	99.59	0.89	100.00	2.88	98.79	1.29
L3DOC(Group2)	98.19	0.88	99.80	2.66	97.39	1.16

7

Figure 4: The test accuracy of each task during training process on ModelNet 10, where test accuracy fluctuation range is limited to $\pm 10\%$ *Average Per-task Accuracy*. Each color corresponds to each task that appears in a continuous way.

4.2.2 Accuracy Evaluation

The performance of all models is presented in Figure 3. Since knowledge in 3D geometric data is more independent than that in neat 2D data, most existing knowledge transfer algorithms do not perform as well as that on 2D data. We thus compare all the competing models with STL-PointNet to show the improvement of 3D classification capabilities.

For the *APA* shown in Figure 3(a), we can have the following observations: **1)** except for the Prog-PointNet, our proposed L3DOC model can achieve the best performance on all datasets, which verifies the impact of improved generalization performance (*i.e., PPA*) via layer-wise point-knowledge factorization and memory attention mechanism. **2)** The reason why Prog-PointNet can obtain the best performance on the three datasets is that it retains all the parameters corresponding to each learned task, *i.e.,* it can ensure the test accuracy for each task is on the corresponding training model. **3)** Comparing with STL-PointNet, most lifelong learning models (*e.g.,* our L3DOC and DF-PointNet) perform better than STL-PointNet in terms of *PPA*, since STL-PointNet cannot transfer effective knowledge from previous tasks to learn the new task. The reason why our L3DOC performs better than DF-PointNet is our L3DOC can capture the unique knowledge of point cloud in different objects, while retaining the global features of point cloud in the task-specific tensor parameters. **4)** For the DEN algorithm, it cannot achieve better performance since it pays more attention to 2D neat data while ignoring the 3D geometry data.

For the *CFR* presented in Figure 3(b), we can find that **1)** the performances of most models have similar trend as that in terms of *APA*, *i.e.,* L3DOC model can achieve the best performance in most cases. **2)** The reason why Prog-PointNet has no catastrophic forgetting is that *APA* is equal to the average task test accuracy under the corresponding task training model.

For the *SC* reported in Figure 3(c), notice that **1)** our proposed L3DOC model is faster than most competing models on

all the three datasets, which is in accord with complexity analysis. **2)** Although Prog-PointNet can achieve the best performance in terms of *APA*, it needs more time to converge since it needs to retrain all the network parameters of the last task. **3)** In the 3DMNIST and ModelNet40 datasets, the *Speed of Convergence* for DF-PointNet, DEN and STL-PointNet have large fluctuations. This is because that the contextual task of the fluctuating position contains objects with similar geometry but in different classes, and neither can overcome the mis-transfer of knowledge. Notice that the number of training convergence in STL-PointNet is too large, which is not indicated in Figure 3(c).

From the Figure 4, it is intuitive to see that both L3DOC(Group1) and L3DOC(Group2) can guarantee efficient test accuracy and less fluctuations on continuous tasks. Although the performance of DF-CNN is significantly improved compared with STL-PointNet (*i.e.*, catastrophic forgetting), it is almost the same on all the tasks.

4.3 Ablation Studies

This subsection studies how the scale parameters and memory attention mechanism affect the performance of our L3DOC models, and the experiment on PointNet++.

4.3.1 Effect of Scale Parameters

For the effect of scale parameters, as shown in Figure 3(a), Figure 3(b) and Figure 3(c), we can notice that our proposed L3DOC(Group1) model with more parameters performs well than L3DOC(Group2) model in terms of the most metrics, but consumes relatively high training time. Meanwhile, this differences in effectiveness and efficiency are more significant on the large-scale ModelNet40 dataset. It is because our L3DOC(Group1) model with more scale parameters can be flexibly adjusted to adapt into different scale tasks, while improving the scalability of the model.
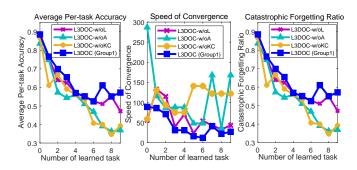

Figure 5: Performance metrics on L3DOC without Memory Attention Mechanism (L3DOC-w/oA), L3DOC's Memory Attention Mechanism without knowledge gap (L3DOC-w/oL, *i.e.*, $\lambda=0$ in Eq. (9)), L3DOC's Memory Attention Mechanism without task-specific tensors gap (L3DOC-w/oKC, *i.e.*, $\{\lambda_{K_i}\}_{i=1}^{l_{max}} = 0$ and $\{\lambda_{C_i}\}_{i=1}^{l_{max}} = 0$ in Eq. (9)), and our L3DOC(Group1)) on ModelNet10.

4.3.2 Effect of Memory Attention Mechanisms

To further examine the efficiency and effectiveness of the memory attention mechanisms, as shown in Figure 5, L3DOC-w/oA, L3DOC-w/oL and L3DOC-w/oKL denote our loss function in Eq. (9) without memory attention mechanisms, λ, and $\{\lambda_{K_i}\}_{i=1}^{l_{max}}$ and $\{\lambda_{C_i}\}_{i=1}^{l_{max}}$, respectively. Notice that the corresponding performances can degrade obviously when removing each modules, which demonstrate the effectiveness of adding memory attention mechanism (*i.e.,* task-specific knowledge among previous tasks), and its rationality on the 3D classification task.

4.3.3 Experimental results on PointNet++

To further illustrate the scalability of L3DOC, we extend our framework into PointNet++ [31] backbone network. As the computational efficiency shown in Table 2, our L3DOC can completes training tasks quickly and in small memory based on its layer-wise point-knowledge factorization architecture on both PointNet and PointNet++ networks. Meanwhile, for the classification accuracy, our L3DOC which is based on PointNet backbone network can achieve a more efficient performance, except for the accuracy metrics: Average-*APA*. It is because the basic PointNet++ network outperforms PointNet in terms of accuracy.

Table 2: Comparison our L3DOC with STL on the performance of average *APA*, average *CFR*, *TT*, computational memory and the number of parameters on ModelNet10. The best performance is indicated in red.

# Backbone Network	PointNet [30]		PointNet++ [31]	
# Framework	STL	L3DOC	STL	L3DOC
Average-*APA*	38.74	60.13	41.66	64.04
Average-*CFR*	0.428	0.684	0.446	0.683
TT (10k sec)	1.306	0.891	34.06	31.57
Memory (1k Mib)	2.435	1.477	2.965	2.481
Parameter (100k)	15.99	9.50	79.89	13.25

5 Conclusion

In this paper, we propose to solve continual 3D object classification task from the perspective of lifelong learning, i.e., Lifelong 3D Object Classification (L3DOC). To achieve lifelong learning, a layer-wise point-knowledge factorization architecture is employed in L3DOC to capture and store the shared point-knowledge, while obtaining more compact parameter representations to reduce computational complexity and memory. To efficiently avoid the catastrophic forgetting, a memory attention mechanism is used to transfer similar task-specific knowledge among different classification tasks. The experiment result on three point cloud benchmark datasets demonstrate the effectiveness and efficiency of our proposed L3DOC model when comparing with the state-of-the-arts.

References

[1] Faisal R. Al-Osaimi. A novel multi-purpose matching representation of local 3d surfaces: A rotationally invariant, efficient, and highly discriminative approach with an adjustable sensitivity. *IEEE Trans. Image Processing*, 25(2):658–672, 2016. 2

[2] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, and et al. Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? *IEEE International Conference on Computer Vision(ICCV)*, pages 2593–2602, 2017. 1

[3] Anders Glent Buch, Lilita Kiforenko, and Dirk Kraft. Rotational subgroup voting and pose clustering for robust 3d object recognition. *IEEE International Conference on Computer Vision (ICCV)*, pages 4137–4145, 2017. 2

[4] Rich Caruana. Multitask learning: A knowledge-based source of inductive bias. *ICML*, pages 41–48, 1993. 2

[5] Long Chen, Hanwang Zhang, Jun Xiao, and et al. SCA-CNN: spatial and channel-wise attention in convolutional networks for image captioning. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6298–6306, 2017. 2

[6] Yang Cong, Dongying Tian, Yun Feng, and et al. Speedup 3-d texture-less object recognition against self-occlusion for intelligent manufacturing. *IEEE Transactions on Cybernetics*, 49(11):3887–3897, 2019. 2

[7] David de la Iglesia Castro. A 3d version of the mnist database of handwritten digits, 2016. https://www.kaggle.com/daavoo/3d-mnist. 6

[8] Jinming Duan, Ghalib Bello, Jo Schlemper, and et al. Automatic 3d bi-ventricular segmentation of cardiac images by a shape-refined multi- task deep learning approach. *IEEE Trans. Med. Imaging*, 38(9):2151–2164, 2019. 1, 2

[9] Yuan Gao, Qi She, Jiayi Ma, and et al. NDDR-CNN: layer-wise feature fusing in multi-task CNN by neural discriminative dimensionality reduction. *CoRR*, abs/1801.08297, 2018. 2

[10] Saurabh Gupta, Pablo Andrés Arbeláez, Ross B. Girshick, and Jitendra Malik. Aligning 3d models to RGB-D images of cluttered scenes. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4731–4740, 2015. 2

[11] Saurabh Gupta, Ross B. Girshick, Pablo Andrés Arbeláez, and Jitendra Malik. Learning rich features from RGB-D images for object detection and segmentation. *Computer Vision - (ECCV)*, 8695:345–360, 2014. 2

[12] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, and et al. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. *11th Asian Conference on Computer Vision (ACCV)*, pages 548–562, 2012. 2

[13] Parisi German Ignacio, Tani Jun, Weber Cornelius, and Wermter Stefan. Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization. *Frontiers in Neurorobotics*, 12, 2018. 2

[14] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. *NIPS*, pages 667–675, 2016. 2

[15] J. Diaz K. Pauwels, L. Rubio and E. Ros. Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2347–2354, 2013. 2

[16] Alexandr A. Kalinin, Ari Allyn-Feuer, Alexander S. Ade, and et al. 3d cell nuclear morphology: Microscopy imaging dataset and voxel-based morphometry classification results. *CVPR Workshops*, pages 2272–2280, 2018. 1

[17] Eunwoo Kim, Chanho Ahn, and Philip H. S. Torr andSonghwai Oh. Deep virtual networks for memory efficient inference of multiple tasks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2710–2719, 2019. 2

[18] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, and et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114(13):3521–3526, 2017. 2

[19] Seungwon Lee, James Stokes, and Eric Eatonr. Learning shared knowledge for deep lifelong learning using deconvolutional networks. *6th Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence(IJCAI)*, pages 2837–2844, 2019. 2, 3, 5, 6, 7

[20] Duohan Liang, Guoliang Fan, Guangfeng Lin, and et al. Three-stream convolutional neural network with multi-task and ensemble learning for 3d action recognition. *CVPR Workshops*, page 0, 2019. 2

[21] Hongsen Liu, Yang Cong, Chenguang Yang, and Yandong Tang. Efficient 3d object recognition via geometric information preservation. *Pattern Recognition*, 92:135–145, 2019. 2

[22] Hao Liu, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Learning deep sharable and structural detectors for face alignment. *IEEE Trans. Image Processing*, 26(4):1666–1678, 2017. 2

[23] Yongxi Lu, Abhishek Kumar, Shuangfei Zhai, and et al. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1131–1140, 2017. 2

[24] C. Wiedemann M. Ulrich and C. Steger. Combining scale-space and similarity-based aspect graphs for fast 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10):1902–1914, 2012. 2

[25] Jorge Armando Mendez Mendez, Shashank Shivkumar, and Eric Eaton. Lifelong inverse reinforcement learning. *NeurIPS*, pages 4507–4518, 2018. 2

[26] Carsten Moenning and Neil A Dodgson. Fast marching farthest point sampling for point clouds and implicit surfaces. *Computer Laboratory Technical Report*, 565:1–12, 20003. 6

[27] Xi Ouyang, Shuangjie Xu, Chaoyun Zhang, and et al. A 3d-cnn and LSTM based multi-task learning architecture for action recognition. *IEEE Access*, 7:40757–40770, 2019. 2

[28] German Ignacio Parisi, Ronald Kemker, Jose L. Part, and et al. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019. 2

[29] Quang-Hieu Pham, Duc Thanh Nguyen, Binh-Son Hua, and et al. JSIS3D: joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8827–8836, 2019. 2

[30] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017. 1, 2, 3, 5, 6, 7, 9

[31] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 5099–5108, 2017. 1, 2, 9

[32] Rajeev Ranjan, Vishal M. Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41:121–135, 2019. 2

[33] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5533–5542, 2017. 2

[34] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. Paris-lille-3d: A point cloud dataset for urban scene segmentation and classification. *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*, pages 2027–2030, 2018. 1

[35] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv: Learning*, 2016. 2, 6, 7

[36] Paul Ruvolo and Eric Eaton. ELLA: an efficient lifelong learning algorithm. *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 28:507–515, 2013. 2, 3

[37] Jan Stria and Václav Hlavác. Classification of hanging garments using learned features extracted from 3d point clouds. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5307–5312, 2018. 1

[38] Gan Sun, Yang Cong, Ji Liu, and et al. Lifelong metric learning. *IEEE Trans. Cybernetics*, 49(8):3168–3179, 2019. 2

[39] Keke Tang, Peng Song, and Xiaoping Chen. 3d object recognition in cluttered scenes with robust shape description and correspondence selection. *IEEE Access*, 5:1833–1845, 2017. 1

[40] Sebastian Thrun. A lifelong learning perspective for mobile robot control. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1:23–30, 1994. 2

[41] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3109–3118, 2015. 2

[42] Zhirong Wu, Shuran Song, and et al. 3d shapenets: A deep representation for volumetric shapes. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015. 2, 6

[43] Kelvin Xu, Jimmy Ba, Ryan Kiros, and et al. Show, attend and tell: Neural image caption generation with visual attention. *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 2048–2057, 2015. 5

[44] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *International Conference on Learning Representations(ICLR)*, 2018. 2, 6, 7

[45] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 70:3987–3995, 2017. 2

[46] Yusen Zhan, Haitham Bou-Ammar, and Matthew E. Taylor. Scalable lifelong reinforcement learning. *Pattern Recognition*, 72:407–418, 2017. 2