Topology-Aware Graph Convolution Network for Few-Shot Incremental 3-D Object Learning

Bingtao Ma, Yang Cong, *Senior Member, IEEE*, and Jiahua Dong

***Abstract*—Three-dimensional (3-D) object recognition has achieved satisfied achievement in both academia and industry. However, most traditional 3-D object classification methods implicitly assume that there are abundant training data from a static distribution. To relax the assumption, we target on a more challenging and realistic setting: few-shot incremental 3-D object learning (FSI3DL), which intends to incrementally classify the new coming 3-D objects with few training data. In order to achieve this, two key challenges need to be concerned: 1) the catastrophic forgetting issue caused by incremental 3-D data with irregular and redundant topological structures and 2) the overfitting issue caused by few-shot training data. To address the first challenge, we use Laplacian spectral analysis based on 3-D meshes to design an embedding network that consists of super-vertex graph convolution (SVGC) module and topology-aware graph attention (TAGA) module. The SVGC is designed to construct the discriminative local topological characteristics for representing the irregular 3-D meshes better. The TAGA is designed to identify redundant topological characteristics. To address the second challenge, a fine-tuning strategy with *model alignment regularization* is investigated. Furthermore, an embedding space selection and fusion (ESSF) strategy is proposed in the inference phase to mitigate catastrophic forgetting and overfitting further. Combining SVGC, TAGA, and alignment regularization with ESSF strategy, a novel topology-aware graph convolution network (TopGCN) is proposed to address the FSI3DL. Experiments on representative 3-D classification datasets validate the superiority of TopGCN.**

***Index Terms*—3-D meshes, class incremental learning, few-shot, graph convolution network (GCN), three-dimensional (3-D) object recognition.**

Manuscript received 2 February 2023; revised 25 May 2023; accepted 20 July 2023. This work was supported in part by the National Science and Technology Major Project of the New Generation of Artificial Intelligence, China, under Grant 2018AAA0102900, and in part by the National Natural Science Foundation of China under Grant 62225310 and Grant 62127807. This article was recommended by Associate Editor Q. Wang. *(Corresponding author: Yang Cong.)*

Bingtao Ma and Jiahua Dong are with the State Key Laboratory of Robotics, Shenyang Institute of Automation, and the Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110016, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: mabingtao93@gmail.com; dongjiahua1995@gmail.com).

Yang Cong is with the State Key Laboratory of Robotics, Shenyang Institute of Automation, and the Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110016, China (e-mail: congyang81@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSMC.2023.3302008.

Digital Object Identifier 10.1109/TSMC.2023.3302008

I. Introduction

THREE-DIMENSIONAL (3-D) object recognition techniques have achieved dramatic success in many applications, such as autonomous vehicle technology [1], intelligent robotics [2], and AR/VR [3]. Representative 3-D object recognition methods can be roughly summarized into five categories: 1) parameterization-based [4]; 2) view-based [5]; 3) voxel-based [6]; 4) point clouds-based [7]; and 5) meshes-based [8]. The successes of these methods are based on an implied assumption that there are abundant training samples from a static distribution.

However, the above assumption can be violated easily in many practical scenarios where new classes of 3-D objects with few training samples arrive incrementally (see Fig. 1). For example, a household cleaning robot or 3-D object classification software is trained on *server-defined* classes (e.g., ModelNet40 [6]). Then, the robot or software is provided to users who may have *user-defined* classes and may add new classes continually in the future. To meet the needs of users, the model underpinning the robot and the software needs to continually learn new *user-defined* classes, while not forgetting the classes learned before. But users are only willing to label very few samples (e.g., 3, 5, or 10) per new class because the sample annotation consumes effort. Therefore, it is critical to equip the model with the ability to learn new classes with few training samples and preserve previous knowledge.

To meet the needs of the above model, we need to solve a challenging problem: few-shot incremental 3-D object learning (FSI3DL), which has two key challenges. First, the new model may forget the knowledge of previously learned classes (i.e., catastrophic forgetting) [9]. Furthermore, the redundant and irregular topological structures of 3-D objects may exacerbate the catastrophic forgetting [10]. The second challenge in FSI3DL is how can we train a robust classifier with few training samples for new classes. A solution is to mine the deeper connection between the new classes and the previous models. Few methods have been proposed to address FSI3DL. Naive solutions are to fine-tune a trained model [11] with new classes, or joint-train a model using all training data from previous and new classes. The former seriously suffers from catastrophic forgetting on previous classes and overfitting on new classes. The latter is impractical because this method is time consuming and resource consuming (e.g., power and storage), and the old training data may not be accessed. Besides, other straightforward solutions are to combine few-shot class incremental learning (FSCIL) methods [12], [13] of image

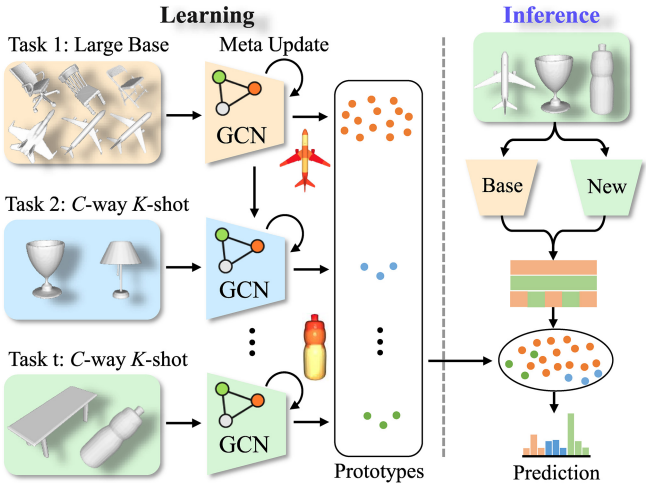

Fig. 1. Illustration of our TopGCN to incrementally learn new classes of 3-D meshes objects on few-shot data (i.e., C-way K-shot tasks). We design an embedding network to learn unique 3-D topological characteristics (e.g., red parts of airplane and bottle) for mitigating catastrophic forgetting. For the inference phase, we build a new embedding space to mitigate catastrophic forgetting and overfitting further.

classification with a 3-D feature extractor [8]. However, they treat each 3-D local characteristic equally to explore 3-D topological characteristics for both past and new classes. Hence, they may fail to neglect redundant topological characteristics and highlight unique topological characteristics for mitigating catastrophic forgetting.

To address FSI3DL, we propose a novel topology-aware graph convolution network (TopGCN) based on meshes. It aims to mitigate catastrophic forgetting and overfitting when incrementally learning new classes of 3-D objects with few training samples. Specifically, to take advantage of the better representation ability of meshes and overcome irregularity and redundancy of meshes, we use Laplacian spectral clustering [14] to design an embedding network that consists of hierarchy super-vertex graph convolution (SVGC) modules and a topology-aware graph attention (TAGA) module. These modules jointly mitigate catastrophic forgetting. Then, to mitigate the overfitting, a fine-tuning strategy with *model alignment* regularization is investigated.

The SVGC modules construct a series of super-vertices with adaptive and larger receptive fields to construct discriminative local topological characteristics, which can characterize 3-D meshes well and overcome irregularity. Then, the TAGA module quantifies the contributions of the above local topological characteristics as illustrated in Fig. 1. In other words, it neglects the redundant topological characteristics with low contributions and focuses on unique 3-D topological characteristics with high contributions. The model alignment regularization aims to align the parameters between the last model and the new model. Hence, combining the fine-tuning (i.e., training the last model by new training samples) with the model alignment regularization, we can transfer some previous knowledge to new classes. Furthermore, motivated by [15], we propose an embedding space selection and fusion (ESSF) strategy. ESSF is implemented in the inference phase and can further mitigate catastrophic forgetting and overfitting. We experimentally show that our method outperforms other SOTAs significantly on ModelNet40 [6], ShapeNet [16], MCB [17], and CO3D [18] datasets.

The contributions of this article are summarized as follows.

1) We design a novel few-shot incremental 3-D learning method based on meshes without saving any exemplars. To the best of our knowledge, this is an early work on few-shot incremental learning in the field of 3-D object recognition.
2) We design SVGC and TAGA modules to construct a novel embedding network that is optimized by a meta-metric loss [19]. The SVGC modules abstract adaptive and larger receptive fields to construct discriminative local topological characteristics, and the TAGA focuses on unique 3-D topological characteristics to mitigate catastrophic forgetting.
3) Combining the fine-tuning strategy with a model alignment regularization, we can mitigate overfitting by transferring the previous knowledge to the few-shot data from new classes.

II. Related Work

A. 3-D Object Classification

With large-scale 3-D object datasets arising, numerous neural networks based on diverse data representations are proposed for 3-D object classification, which can be divided into five categories [20], [21]. *Parameterization-based methods* [4], [22] flatten a 3-D meshes object to a two-dimensional (2-D) image. Although they also enjoy common benefits derived from CNNs, their performance needs to be improved. *View-based methods* [5], [23] collect a group of multiview images for a 3-D object, then apply 2-D CNN and view pooling to extract the global features. Although they are effective, they suffer from self-occlusion and determining the number of views. *Voxel-based methods* [6], [24] convert a 3-D object to voxel grids and apply 3-D CNNs to extract features. They are limited by discretion errors and exponentially growing computational costs. Recently, sparse methods [25] and Octrees methods [26] are proposed to improve efficiency. *Point Clouds-Based Methods:* PointNet [27] learns pointwise features from disordered point clouds directly, while it ignores the local information. PointNet++ [7] solves this issue by aggregating neighbors. They have inspired some other works on point clouds [28], [29]. *Meshes-based methods* aim utilize the better representation ability of meshes and overcome the irregularity of meshes. They regard vertices [30], edges [11], [31], or faces [8] as the unit of extracting features. For example, MeshNet [8] regards polygon faces as the unit and designs some blocks to capture and aggregate face features. However, these methods are static and cannot tackle the practical FSI3DL scenario well.

B. Few-Shot Class Incremental Learning

Few-shot learning (FSL) aims to train a model to classify unseen classes with a few training data. Most research works in FSL are meta-learning based [32], [33] or metric-learning

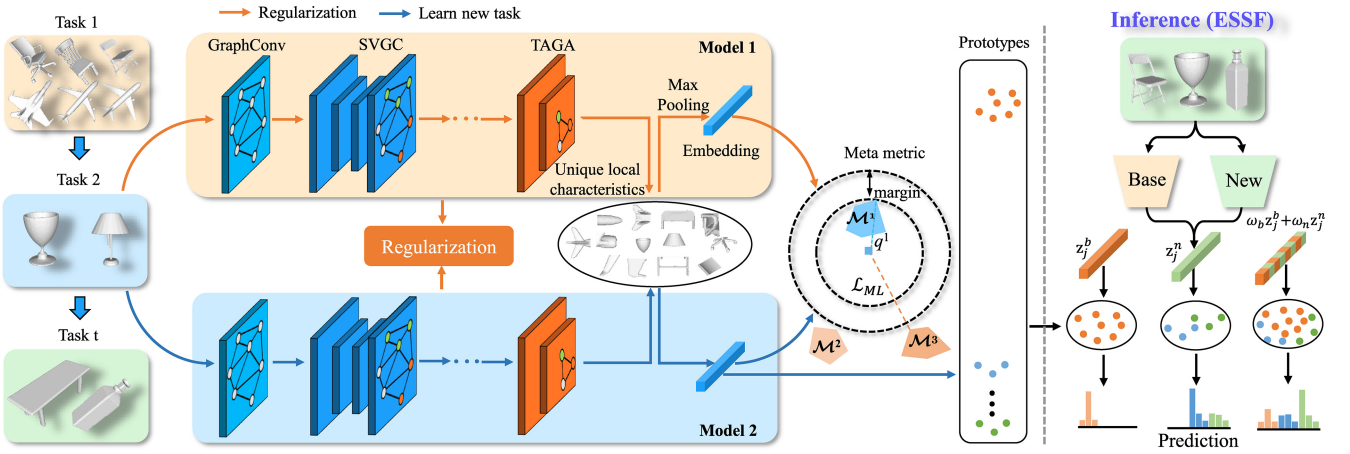

Fig. 2. Overview of our TopGCN method, which mainly consists of stacked SVGC modules to construct discriminative local topological characteristics while achieving adaptive and larger receptive field, a TAGA module explores what local topological characteristics are beneficial for incremental learning. The embedding network is optimized by a model alignment regularization loss term and a meta-metric loss term. In the inference phase, an ESSF strategy selects proper embedding to further mitigate catastrophic forgetting and overfitting. Finally, we classify all seen classes according to the selected embedding by the nearest class mean classifier.

based [34], [35]. Class-incremental learning (CIL) [36], [37], [38], [39] is an active machine learning task that learns a unified classifier continually to recognize all encountered classes and tries to mitigate catastrophic forgetting. There are many CIL works in image classification, but few works in 3-D object classification [10]. Although I3DOL [10] performs well on point clouds data, it cannot be extended to meshes data. In addition, these CIL works are designed for new classes with many training data, and most CIL methods store many exemplars of old classes.

Recently, FSCIL, as a more challenging and practical problem, is proposed by TOPIC [12]. Until now there are not many related works [13], [40], [41]. TOPIC [12] utilizes a neural gas network [42], [43] to keep the topological structure of the feature manifold for preserving old knowledge while introducing new classes incrementally. Cheraghian et al. [13] proposed a semantically guided knowledge distillation method by using a visual-semantic alignment strategy and does not make network parameters increase while learning new classes incrementally. IDLVQ [40] proposes a nonparametric method to balance preserving old knowledge and learning new knowledge based on learning vector quantization (LVQ) [44] in embedding space. Although IDLVQ [40] achieves SOTA performance on image classification. These methods in 2-D vision cannot be employed to 3-D object recognition directly because the irregular and redundant topological structures of meshes representation make them difficult to explore what 3-D topological characteristics are beneficial for FSI3DL. Our method outperforms IDLVQ [40] with a large margin and does not save any exemplars.

III. Few-Shot Incremental 3-D Object Learning

In this section, we first state the FSI3DL problem definition and the overall framework of our proposed TopoGCN method. Then, we introduce detailed formulations of our method.

A. Problem Definition and Overview

FSI3DL has a stream of disjoint tasks $\mathcal{D} = \{\mathcal{D}^1, \ldots, \mathcal{D}^T\}$, where $\mathcal{D}^t = \{\mathcal{D}^t_{tr}, \mathcal{D}^t_{te}\}$ corresponds to the t-th incremental task. $\mathcal{D}^t_{tr}$ is the training set of task t and $\mathcal{D}^t_{te}$ is the test set of task t. $\{\mathcal{G}^t_j, y^t_j\} \in \mathcal{D}^t$ is a pair of 3-D meshes object $\mathcal{G}^t_j$ and its label $y^t_j \in \mathcal{C}^t$. $\mathcal{G}^t_j = \{\mathcal{V}, \mathcal{A}\}$ can be regarded as a graph, where $\mathcal{V} = \{v_i\}^N_i$ contains N vertices and each vertex is defined by its 3-D coordinate, and $\mathcal{A} \in \mathbb{R}^{N\times N}$ is an adjacency matrix. $\mathcal{C}^t = \{c^t_1, \ldots, c^t_{|\mathcal{C}^t|}\}$ is the set of classes of $\mathcal{D}^t$. The classes of different tasks are disjoint, i.e., $\forall i, j,\ \mathcal{C}^i \cap \mathcal{C}^j = \varnothing$, where $i \neq j$. Base task $\mathcal{D}^1$ (i.e., task 1) contains many base classes with many training samples, and remaining each new task $\mathcal{D}^t$ $(t > 1)$ contains C classes and each class has K training samples. This setting is defined as C-way K-shot FSCIL setting [12]. The model is incrementally trained on $\mathcal{D} = \{\mathcal{D}^1, \ldots, \mathcal{D}^T\}$, while only training samples from $\mathcal{D}^t$ are accessible during the t-th training task. Afterward, the model is evaluated on all the learned classes.

Fig. 2 illustrates the overview of our method. First, we apply a graph convolution on a 3-D mesh object $\mathcal{G}$ (please note that we omit scripts t and j for ease of readability) to get initial local topological characteristics. Then, stacked SVGC modules further incorporate local topological characteristics on $\mathcal{G}$ and super-vertex graphs $\{\mathcal{SG}_l\}^L_{l=1}$ that are obtained by Laplacian spectral clustering. Afterward, a topology-aware attention module explores what 3-D topological characteristics in local topological structures are beneficial to mitigate catastrophic forgetting. Specifically, it stresses unique 3-D topological characteristics while neglecting common topological characteristics. The above modules form an embedding network that is optimized by a model alignment regularization loss term and a meta-metric loss term. The model alignment regularization loss aims to mitigate overfitting by aligning the parameters between the last model and the new model. After training an embedding network at task t, we compute prototypes of $\mathcal{D}^t$. Furthermore, an ESSF strategy is used in the inference phase, which selects proper embedding to further

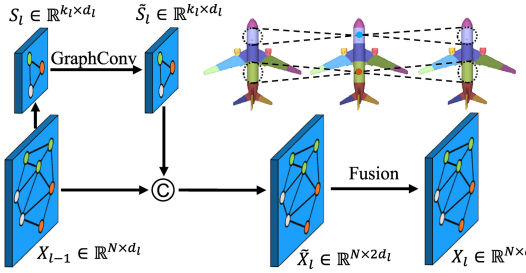

Fig. 3. Detailed demonstration of SVGC. The different patches of the airplane represent different super-vertices.

mitigate catastrophic forgetting and overfitting. Finally, we employ the nearest class mean classifier to classify all the learned classes.

B. Super-Vertex Graph Convolution

A 3-D meshes object is composed of some main local topological structures (e.g., an airplane mainly consists of wing, engine, tail, head, etc.) that can be used to construct discriminative local topological characteristics [10]. Most previous methods [8], [11] design various graph convolution networks (GCNs) on 3-D meshes to extract local characteristics from a small neighborhood (e.g., the 1-ring neighbors). However, a small neighborhood has a smaller receptive field when the network is shallow, which is not beneficial for constructing discriminative local topological characteristics [45] and can not explore the main local topological structures.

To explore the main local topological structures and construct discriminative local topological characteristics, we use Laplacian spectral clustering [14] and graph convolution to design an SVGC module that can be stacked hierarchically. As shown in Fig. 3, the Laplacian spectral clustering can produce some super-vertices that represent the main local topological structures. For example, the different patches of the airplane in Fig. 3 represent different super-vertices. Simultaneously, combined with graph convolution, we can abstract adaptive and larger local topological regions (i.e., receptive field) along the hierarchy to construct discriminative local topological characteristics. As shown in Fig. 3, SVGC is a U-net-like structure and has three key layers: 1) a mesh pooling layer; 2) a super-vertex convolution layer; and 3) a fusion layer.

1) Mesh Pooling Layer: Laplacian matrix of a mesh object $\mathcal{G}$ is computed as $H = A^{-1}(D - W)$ [46], where $A = \mathrm{diag}(a_1, \ldots, a_N)$ represent vertex weights, a_i is local Voronoi area that is equivalent to one-third of the total area of 1-ring triangles, D is a diagonal degree matrix with entry $d_i = \sum_{j=1}^{N} w_{ij}$, and the sparse cotangent weight matrix $W \in \mathbb{R}^{N \times N}$ contains cotangent term w_{ij} that is defined as [46]

$$w_{ij} = \begin{cases} 0.5\big(\cot \alpha_{ij} + \cot \beta_{ij}\big), & j \in \mathcal{N}_i \\ 0, & i = j \text{ or } j \notin \mathcal{N}_i \end{cases} \tag{1}$$

where $\mathcal{N}_i$ is the 1-ring topological neighborhood of vertex v_i, and α_{ij} and β_{ij} denote the two angles opposite of edge (i, j). We perform Laplacian spectral clustering [14] on Laplacian matrix H to cluster similar vertices (i.e., the same color vertices of X_{l-1} in Fig. 3) into a patch that is a main local topological structure and is denoted as a super-vertex. Then, we conduct graph convolution on super-vertices to extract discriminative local topological characteristics.

We use Laplacian spectral clustering to produce k_l super-vertices $\mathcal{SV}_l = \{sv_{l,i}\}_{i=1}^{k_l}$ and a cluster mask $M_l = \{m_i^l | m_i^l \in \{1, 2, \ldots, k_l\}\}_{i=1}^{N}$ in hierarchy level l, where an entry m_i^l indicates that vertex v_i belongs to super-vertex sv_{l,m_i^l}. $l = 1, 2, \ldots, L$, and L represents the number of hierarchy levels. As shown in Fig. 3, we build a super-vertex graph $\mathcal{SG}_l$ by connecting two patches/super-vertices if original graph $\mathcal{G}$ has at least one edge between two patches (e.g., the green and orange vertices).

Super-vertex feature is obtained by applying a channel-wise max-pooling operation on the input features of all mesh vertices that belong to the super-vertex $sv_{l,j}$. Taking max-pooling as an example, the super-vertex feature $\boldsymbol{s}_{l,j} \in \mathbb{R}^{d_l}$ of the lth SVGC module is formulated as

$$\boldsymbol{s}_{l,j} = \max_{m_i^l = j} \boldsymbol{x}_{l-1,i} \tag{2}$$

where $\boldsymbol{x}_{l-1,i}$ is the vertex feature of the original graph and is collected into matrix $X_{l-1} = \{\boldsymbol{x}_{l-1,i}\}_{i=1}^{N} \in \mathbb{R}^{N \times d_l}$.

2) Super-Vertex Convolution Layer: With adaptive and large receptive field, we can construct more discriminative local topological characteristics [45]. Hence, to obtain adaptive and large receptive field, we perform graph convolutions on $\mathcal{SG}_l$ to aggregate topological characteristics across super-vertices as for example in EdgeConv [28]. Specifically, the output feature $\tilde{\boldsymbol{s}}_{l,i} \in \mathbb{R}^{d_l}$ of super-vertex $sv_{l,i}$ is computed as

$$\tilde{\boldsymbol{s}}_{l,i} = \max_{j \in \mathcal{N}_{l,i}} \mathrm{LeakyRelu}\big(F_{l,s}\big(\boldsymbol{s}_{l,i} © \big(\boldsymbol{s}_{l,j} - \boldsymbol{s}_{l,i}\big); \theta_{l,s}\big)\big) \tag{3}$$

where $\mathcal{N}_{l,i}$ is the 1-ring topological neighbors of super-vertex $sv_{l,i}$, $F_{l,s} : \mathbb{R}^{2d_l} \to \mathbb{R}^{d_l}$ is a shared nonlinear function implemented as an MLP with trainable parameters $\theta_{l,s}$ and © is the concatenation operation.

3) Fusion Layer: We note that continually performing graph convolution on the super-vertices may lose some detailed information. As shown in Fig. 3, to preserve more detailed information, we concatenate the vertex feature $\boldsymbol{x}_{l-1,i} \in \mathbb{R}^{d_l}$ and the corresponding super-vertex feature $\tilde{\boldsymbol{s}}_{l,m_i^l}$, i.e., $\tilde{\boldsymbol{x}}_{l,i} = \tilde{\boldsymbol{s}}_{l,m_i^l} © \boldsymbol{x}_{l-1,i}$. Then, we apply the EdgeConv to get final output

$$\boldsymbol{x}_{l,i} = \max_{j \in \mathcal{N}_i} \mathrm{LeakyRelu}\big(F_{l,f}\big(\tilde{\boldsymbol{x}}_{l,i} © \big(\tilde{\boldsymbol{x}}_{l,j} - \tilde{\boldsymbol{x}}_{l,i}\big); \theta_{l,f}\big)\big) \tag{4}$$

where $\mathcal{N}_i$ represents the 1-ring topological neighborhood of vertex v_i, $F_{l,f} : \mathbb{R}^{4d_l} \to \mathbb{R}^{d_l}$ with trainable parameters $\theta_{l,f}$ and the other symbols are similar to (3). These output topological characteristics flow into a subsequent module.

C. Topology-Aware Graph Attention

Although our SVGC could construct super-vertices adaptively with discriminative local topological characteristics, each super-vertex contributes equally to exploring discriminative 3-D topological characteristics for incremental learning, which is not beneficial for incremental learning. In other words, some super-vertices representing unique 3-D characteristics can help mitigate catastrophic forgetting of old classes,

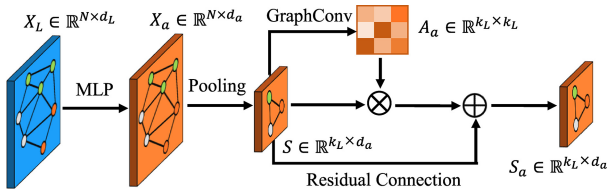

Fig. 4. Detailed demonstration of TAGA.

but the others representing common characteristics can promote catastrophic forgetting. Thus, as shown in Fig. 4, we design a TAGA module to stress unique 3-D topological characteristics while neglecting common topological characteristics. Specifically, the uniqueness of a super-vertex is decided by itself and the difference between itself and its 1-ring topological neighbors. Thus, we use graph convolution and residual mechanism to construct the TAGA module. The output $X_L = \{\boldsymbol{x}_{L,i}\}_{i=1}^{N} \in \mathbb{R}^{N\times d_L}$ of the last SVGC module is processed as

$$\boldsymbol{x}_{a,i} = \text{LeakyRelu}\big(F_{a,1}\big(\boldsymbol{x}_{L,i};\theta_{a,1}\big)\big) \tag{5}$$

where $F_{a,1} : \mathbb{R}^{d_L} \to \mathbb{R}^{d_a}$ is a shared MLP function with trainable parameters $\theta_{a,1}$. The TAGA module has the same number of super vertices as the last SVGC module. Then, we apply the mask M_L on $X_a = \{\boldsymbol{x}_{a,i}\}_{i=1}^{N} \in \mathbb{R}^{N\times d_a}$ to get super-vertices features $S = \{\boldsymbol{s}_i\}_{i=1}^{k_L} \in \mathbb{R}^{k_L\times d_a}$ as $\boldsymbol{s}_{l,j}$. Then, quantified super-vertex feature is computed as

$$\boldsymbol{s}_{a,i} = \sum_{j\in\mathcal{N}_{a,i}} a_{ij}\boldsymbol{e}_{ij} + \boldsymbol{s}_i \tag{6}$$

where $\boldsymbol{e}_{ij} = \boldsymbol{s}_i - \boldsymbol{s}_j$ is edge feature that represents the difference between two super-vertices, and $A_a = \{a_{ij}\}_{i,j=1}^{k_L} \in \mathbb{R}^{k_L\times k_L}$ are attention weights. The larger $\boldsymbol{e}_{ij}$ produces a larger attention weight which means super-vertex sv_i is a more unique structure than others and makes a greater contribution to mitigating catastrophic forgetting. The a_{ij} is computed as

$$a_{ij} = \text{Sigmod}\left(\text{Relu}\left(F_{a,2}\left(e_{ij};\theta_{a,2}\right)\right)\right) \tag{7}$$

where $F_{a,2} : \mathbb{R}^{d_a} \to \mathbb{R}$ is a shared nonlinear function implemented as an MLP with trainable parameters $\theta_{a,2}$. Finally, we utilize a channel-wise max-pooling operation on $S_a = \{\boldsymbol{s}_{a,i}\}_{i=1}^{k_L} \in \mathbb{R}^{k_L\times d_a}$ to obtain the global embedding $\mathbf{z} = f(\mathcal{G}) \in \mathbb{R}^{d_a}$ of a 3-D meshes object, where $f(\cdot)$ represents the network model.

D. Embedding Network Learning

The softmax-based classifier is challenging and harder for incremental learning [47], and this situation becomes worse when the training data of new classes are few. Therefore, we focus on the combination of embedding network learned with metric-learning loss and an NCM classifier [37]. The objective function is formulated as

$$\mathcal{L} = \lambda\mathcal{L}_{\text{EWC}} + \mathcal{L}_{\text{MML}} \tag{8}$$

where $\mathcal{L}_{\text{EWC}}$ is a model alignment regularization loss for constraint parameters changeless [47], [48], $\mathcal{L}_{\text{MML}}$ is a metric-learning loss, and λ is the tradeoff parameter. $\mathcal{L}_{\text{EWC}}$ is formulated as

$$\mathcal{L}_{\text{EWC}} = \sum_p \frac{1}{2}\mathcal{F}_p^{t-1}\left(\theta_p^t - \theta_p^{t-1}\right)^2 \tag{9}$$

where $\mathcal{F}_p^{t-1}$ is the Fisher information matrix [48] that is got at task $t-1$, θ_p^t and θ_p^{t-1} are the network parameters of task t and task $t-1$, respectively. $\mathcal{L}_{\text{EWC}}$ sums over all parameters θ_p of the network.

In previous methods [47], [48], this regularization term is used to keep previous knowledge for mitigating catastrophic forgetting. However, we have a new find in this article: $\mathcal{L}_{\text{EWC}}$ can also be used to mitigate the overfitting. The reason is that $\mathcal{L}_{\text{EWC}}$ reduces the difference between the new and last model, which can transfer some effective knowledge from the last model to the current new model. Then, with the aid of transferred knowledge, the new model can achieve a better generalization with few data (i.e., mitigating the overfitting).

$\mathcal{L}_{\text{MML}}$ is the meta-metric loss [19] that optimizes the set-based distance in a meta way to better learn base or new classes. Specifically, at task t, to construct a batch similar to the form of meta-learning, a support set $\mathcal{S}^t = \{s_i^c | c = c_r^t, i = 1,\ldots,n_s\}_{r=1}^{|\mathcal{C}^t|}$ and a query set $\mathcal{Q}^t = \{q_i^c | c = c_r^t, i = 1,\ldots,n_q\}_{r=1}^{|\mathcal{C}^t|}$ are randomly sampled from the training set $\mathcal{D}_{tr}^t$. The support samples of class $c \in \mathcal{C}^t$ form a meta cell $\mathcal{M}^c = \{s_i^c | i = 1,\ldots n_s\}$. As shown in Fig. 2, for each query sample $q_j^{c'}$ of $\mathcal{Q}^t$, we aim to minimize its distance to the positive meta-cell $\mathcal{M}^{c'}$ and push away other negative meta-cells $\mathcal{M}^c$ $(c \neq c')$. $\mathcal{L}_{\text{MML}}$ is formulated as

$$\mathcal{L}_{\text{MML}} = \sum_c \sum_j^{n_q} \log\left(1 + \sum_{c\neq c'} \exp\left(d_j^{c'} - d_j^c + m\right)\right) \tag{10}$$

where m is a margin parameter, $d_j^{c'}$ denotes the farthest intraclass distance between $q_j^{c'}$ and the corresponding positive meta-cell $\mathcal{M}^{c'}$, and d_j^c denotes the nearest interclass distance between $q_j^{c'}$ and negative meta-cell $\mathcal{M}^c$. This process is called as hard mining strategy [19]

$$d_j^c = d\left(q_j^{c'}, \mathcal{M}^c\right) = \begin{cases} \max_i\left(d\left(f\left(q_j^{c'}\right), f\left(s_i^c\right)\right)\right), & c' = c \\ \min_i\left(d\left(f\left(q_j^{c'}\right), f\left(s_i^c\right)\right)\right), & c' \neq c \end{cases} \tag{11}$$

where $d(\cdot,\cdot)$ is a distance metric function.

After training an embedding network with the optimization of (8) at task t, we compute prototype $\boldsymbol{\mu}_c$ [37] of class $c \in \mathcal{C}^t$ (please note that we omit script t for ease of readability)

$$\boldsymbol{\mu}_c = \frac{1}{n_c}\sum_j [y_j = c]\mathbf{z}_j^n \tag{12}$$

where $\mathbf{z}_j^n = f^t(\mathcal{G}_j)$ is the embedding of a training sample, class c has n_c training samples, and $[y_j = c]$ is 1 if $y_j = c$ is true and 0 otherwise. The prototype $\boldsymbol{\mu}_c$ $(c \in \mathcal{C}^t)$ is computed in task t, since task $t+1$ cannot access the training data of old tasks. Then, the NCM classifier [37] is applied on the embedding space of $f^t(\cdot)$ to classify all seen classes as follow:

$$\hat{y}_j = \underset{c\in\mathcal{C}_s}{\text{argmin}}\ \text{dist}\left(\mathbf{z}_j^n, \boldsymbol{\mu}_c\right). \tag{13}$$

E. Embedding Space Selection and Fusion

To further mitigate the catastrophic forgetting and overfitting, we propose an ESSF strategy in the inference phase. The ESSF strategy selects the base, new, or fused embeddings of the test data of all seen classes. Then, we modify the NCM classifier to classify these selected embeddings. As illustrated in the right part of Fig. 2, we extract base embedding $\mathbf{z}_j^b = f^1(\mathcal{G}_j)$ and new embedding $\mathbf{z}_j^n = f^t(\mathcal{G}_j)$ of a test sample from the base model $f^1(\cdot)$ and latest model $f^t(\cdot)$, respectively. Then, we select the base, new, or fused embedding as follows:

$$\mathbf{z}_j = \begin{cases} \mathbf{z}_j^b, & \omega_j^b - \omega_j^n > \tau \\ \mathbf{z}_j^n, & \omega_j^n - \omega_j^b > \tau \\ \omega_j^b \mathbf{z}_j^b + \omega_j^n \mathbf{z}_j^n, & |\omega_j^n - \omega_j^b| \le \tau \end{cases} \tag{14}$$

where ω_j^b and ω_j^n represent the possibility that $\mathcal{G}_j$ come from the base and new classes, respectively, and $\tau \in (0, 1)$ is a predefined threshold. If $\omega_j^b - \omega_j^n > \tau$, $\mathcal{G}_j$ is more likely to come from the base classes $\mathcal{C}^1$. Conversely, $\mathcal{G}_j$ is more likely to come from the new classes $\mathcal{C}_n = \bigcup_t \mathcal{C}^t$ $(t > 1)$ when $\omega_j^n - \omega_j^b > \tau$. However, the source of $\mathcal{G}_j$ is unclear when $|\omega_j^n - \omega_j^b| \le \tau$, then we use an fusion of base embedding $\mathbf{z}_j^b$ and novel embedding $\mathbf{z}_j^n$. The possibility of ω_j^b and ω_j^n is computed as

$$\omega_j^b = \frac{\exp\left(-d\left(\mathbf{z}_j^b, \boldsymbol{v}_j^b\right)\right)}{\exp\left(-d\left(\mathbf{z}_j^b, \boldsymbol{v}_j^b\right)\right) + \exp\left(-d\left(\mathbf{z}_j^n, \boldsymbol{v}_j^n\right)\right)}, \omega_j^n = 1 - \omega_j^b \tag{15}$$

where $\boldsymbol{v}_j^b = \sum_{i \in \mathcal{C}^1} \alpha_{ji} \boldsymbol{\mu}_i$ and $\boldsymbol{v}_j^n = \sum_{i \in \mathcal{C}_n} \beta_{ji} \boldsymbol{\mu}_i$ denote the base and new embedding space center, respectively. α_i denotes the weight value of the ith base prototype and β_i is also similar. They are defined as follows:

$$\alpha_{ji} = \frac{\exp\left(-d\left(\mathbf{z}_j^b, \boldsymbol{\mu}_i\right)\right)}{\sum_{c \in C^1} \exp\left(-d\left(\mathbf{z}_j^b, \boldsymbol{\mu}_c\right)\right)}$$
$$\beta_{ji} = \frac{\exp\left(-d\left(\mathbf{z}_j^n, \boldsymbol{\mu}_i\right)\right)}{\sum_{c \in C_n} \exp\left(-d\left(\mathbf{z}_j^n, \boldsymbol{\mu}_c\right)\right)}. \tag{16}$$

Finally, to classify the selected embedding $\mathbf{z}_j$ from (14), we modify the NCM classifier [37] as follows:

$$\hat{y}_j = \underset{c \in \mathcal{C}'}{\operatorname{argmin}} \operatorname{dist}\left(\mathbf{z}_j, \boldsymbol{\mu}_c\right)$$
$$\mathcal{C}' = \begin{cases} \mathcal{C}^1, & \omega_j^b - \omega_j^n > \tau \\ \mathcal{C}_n = \bigcup_{k=2}^t \mathcal{C}^k, & \omega_j^n - \omega_j^b > \tau \\ \mathcal{C}_s = \bigcup_{k=1}^t \mathcal{C}^k, & |\omega_j^n - \omega_j^b| \le \tau. \end{cases} \tag{17}$$

If $\omega_j^b - \omega_j^n > \tau$, we classify $\mathbf{z}_j = \mathbf{z}_j^b$ only within base classes $\mathcal{C}^1$. Conversely, we classify $\mathbf{z}_j = \mathbf{z}_j^n$ only within new classes $\mathcal{C}_n$. If $|\omega_j^n - \omega_j^b| \le \tau$, we classify $\mathbf{z}_j = \omega_j^b \mathbf{z}_j^b + \omega_j^n \mathbf{z}_j^n$ within all seen classes $\mathcal{C}_s$.

The base embedding space of $f^1(\cdot)$ performs well on the large-scale base classes and has a sound ability of feature extraction that is helpful to mitigate the overfitting, despite its poor results on unseen new classes. On the contrary, the incremental-learning embedding space of $f^t(\cdot)$ performs poorly on the base classes and performs well on new classes. Equation (14) chooses different embedding depending on possibilities ω_j^b and ω_j^n, it combines the advantages of each other and overcomes the weakness of each other. The optimization and evaluation pipeline is summarized in Algorithm 1.

Algorithm 1 Optimize and Evaluate Our Method (Task t)

Input: Training data $\mathcal{D}_{tr}^t$ of task t, test data $\mathcal{D}_{te} = \bigcup_{i=1}^t \mathcal{D}_{te}^i$, model $f^{t-1}(\cdot)$ when $t > 1$, model $f^1(\cdot)$ when $t > 2$, old prototypes $\{\boldsymbol{\mu}_c | c \in \mathcal{C}_p, \mathcal{C}_p = \bigcup_{i=1}^{t-1} \mathcal{C}^i\}$ when $t > 1$, the number of query samples each class $n_q = 1$, the number of support samples each class $n_s = K - n_q$, and margin parameter m;

Output: New model $f^t(\cdot)$, new prototypes $\{\boldsymbol{\mu}_c | c \in \mathcal{C}^t\}$;

```
if t = 1 then
    Initialize f^1(·) randomly;
    for epoch = 1, 2, ..., 100 do
        for Randomly sample [S^1, Q^1] in D_tr^1 do
            Optimize f^1(·) by Eq. (10);
        end for
    end for
    Compute prototypes on D_tr^1 by Eq. (12);
    Evaluate our method on D_te^1 by Eq. (13).
else
    Initialize f^t(·) with f^{t-1}(·);
    for epoch = 1, 2, ..., 30 do
        Randomly resample [S^t, Q^t] in D_tr^t;
        Optimize f^t(·) by Eq. (8);
    end for
    Compute new prototypes on D_tr^t by Eq. (12);
    Evaluate our method on D_te by Eq. (17).
end if
```

IV. Experiments

In this section, following 2-D FSCIL methods in image classification field (e.g., TOPIC [12] and IDLVQ [40]), we conduct comprehensive experiments on widely used 3-D object recognition datasets: ModelNet40 [6], ShapeNet [16], MCB [17], and CO3D [18].

A. Datasets

ModelNet40 [6] has 9843 training samples and 2468 test samples from 40 widely varied classes of 3-D meshes objects. We choose 16 classes as the base task (i.e., task 1), whose all training samples are utilized to optimize the base model. The remaining 24 classes are split into eight new few-shot incremental learning tasks. Each new task contains three new classes and each new class has five randomly selected training samples (i.e., *3-way 5-shot* setting). The test set remains unchanged from the original dataset. There are nine training tasks in total.

ShapeNet [16] contains 35 037 training samples and 5053 validation samples from 55 widely varied classes of 3-D meshes objects. We use 53 classes of them and choose 21 classes as base task, whose all training samples are utilized to optimize the base model. The remaining 32 classes are split

TABLE I
SETTINGS OF DIFFERENT DATASETS

Dataset	Classes	Base Classes	New Task	Tasks
ModelNet40 [6]	40	16	3-way 5-shot	9
ShapeNet [16]	53	21	4-way 5-shot	9
MCB [17]	68	18	5-way 5-shot	11
CO3D [48]	50	25	5-way 5-shot	6

into eight new few-shot incremental learning tasks. Each new task contains four new classes and each new class has five randomly selected training samples (i.e., *4-way 5-shot*). The original validation is treated as the test set. There are nine training tasks in total.

MCB [17] contains 46 978 training samples and 11 716 test samples from 68 classes of 3-D meshes mechanical components. We choose 18 classes as base task, whose all training samples are utilized to optimize the base model. The remaining 50 classes are split into ten new few-shot incremental learning tasks. Each new task contains 5 new classes and each new class has five randomly selected training samples (i.e., *5-way 5-shot*). The test set remains unchanged from the original dataset. There are 11 training tasks in total.

CO3D [18] is a realistic dataset. It contains 16 557 training samples and 1732 test samples from 50 MS-COCO classes of 3-D point cloud objects with severe noise. We choose 25 classes as base task, whose all training data are used to train the base model. The remaining 25 classes are treated as new classes and split into five new few-shot incremental learning tasks. Each new task contains five new classes and each new class has five randomly selected training samples (i.e., *5-way 5-shot*). The test set remains unchanged from the original dataset. There are six training tasks in total.

We summarize the settings of different datasets in Table I. Meshes objects in ModelNet40, ShapeNet, and MCB datasets are not manifold meshes, which limits us perform spectral clustering well. To address this issue, we use the method proposed by [54] to convert original meshes into watertight manifold meshes, and simplify them to 1024 vertices by mesh decimation [55]. We reconstruct point cloud objects of the CO3D dataset into meshes objects and simplify them to 1024 vertices by mesh decimation.

B. Implementation Details

1) Training Details: We utilize PyTorch to implement our method and use Adam [56] to optimize the model. Momentum is 0.9 and weight decay is 0.0001. For the base task, the model is trained for 100 epochs. The initial learning rate is 0.001 and is decayed by 0.5 every 20 epochs. We train our model with a learning rate of 0.0001 for 30 epochs in each new task. For the meta-metric loss, each class has one query sample (i.e., $n_q = 1$), and the number of support data per class $n_s = K - n_q$. The support set $\mathcal{S}^t$ and query set $\mathcal{Q}^t$ are randomly sampled in each batch, where the batch size is $|\mathcal{C}^t| \times (n_s + n_q)$. For the architecture of the network, the first GraphConv layer of Fig. 2 is an EdgeConv layer with one shared fully connected layer (64). $F_{l,s}$, $F_{l,f}$, $F_{a,1}$, and $F_{a,2}$ are implemented by one shared fully connected layer, respectively. We use two SVGC modules (i.e., $L = 2$), per SVGC has 16 super-vertices (i.e., $k_1 = k_2 = 16$) and set $d_1 = d_2 = 64$ and $d_a = 512$. In other words, the output dimensions of $F_{l,s}$ and $F_{l,f}$ are 64, the output dimension of $F_{a,1}$ is 512, and the input and output dimensions of $F_{a,2}$ are 512 and 1, respectively. The final embedding is 512 dimensions and is normalized. We adapt squared Euclidean distance $d(\mathbf{z}, \mathbf{z}') = ||\mathbf{z} - \mathbf{z}'||_2^2$ as the distance metric function. We set the threshold $\tau = 0.2$ as the default. We augment the data during the training by jittering the vertex positions via a Gaussian noise with a mean of zero and a standard deviation of 0.01 as in MeshNet [8].

2) Comparative Methods: For comparative experiments, we compare our method with the fine-tune and joint train methods. The former only fine-tunes the embedding network of task $t-1$ at current task t with the training data of $\mathcal{D}^t$. The latter trains the embedding network of task $t-1$ at task t with all training data $\bigcup_{i=1}^{t} \mathcal{D}_{te}^i$. They are optimized by triplet loss [57]. We compare some representative CIL methods in our FSI3DL setting, including the classical iCaRL [37] and the SOTA methods NCM [50], EEIL [49], and SDC [47]. We also compare our method with IDLVQ [40], FACT [53], and C-FSCIL [52] that are the SOTA FSCIL methods of image classification. For a fair comparison, these methods utilize modified DGCNN [28] as the embedding network. We replace its input with meshes and replace its KNN neighbors with 1-ring neighbors of meshes. The final embedding is 512 dimensions and is normalized. The rest is consistent with DGCNN [28]. We also compare with one recently proposed 3-D object classification algorithm PAConv [51]. We modify PAConv [51] to the FSI3DL task by using the knowledge distillation loss of iCaRL [37] to mitigate PAConv's catastrophic forgetting. Except for IDLVQ [40] saves one exemplar per old class, there are no exemplar data available for our method and other competing methods. All the above methods are evaluated on all the seen classes $\mathcal{C}_s = \bigcup_{i=1}^{t} \mathcal{C}^i$ in the inference phase.

3) Evaluation Protocol: After training a network on D^t, we evaluate the model on all the encountered classes and compute the *average incremental accuracy* [58], [59]. We repeat the whole learning process ten times on each dataset and report the average test accuracy. We also report *average forgetting* [60] to evaluate the forgetting of previous tasks. In task k, it quantifies forgetting for the jth task $f_j^k = \max_{l \in 1,\ldots,k-1}(a_{l,j} - a_{k,j})\ \forall j < k$ where $a_{l,j}$ is the accuracy of task j after training task l. The average forgetting at the kth task is defined as $F_k = [1/(k-1)] \sum_{j=1}^{k-1} f_j^k$.

C. Comparison With Comparative Methods

1) ModelNet40 Results: We present the accuracy comparisons of few-shot incremental learning on the ModelNet40 dataset in Table II. We summarize the observations from Table II as follows:

1) Comparing the performance of different methods at task 1, our method outperforms the others by at least 1.12%, which demonstrates that the irregular meshes can be characterized well by our embedding network (SVGC + TAGA) and meta-metric loss $\mathcal{L}_{\text{MML}}$. Furthermore, our TopGCN boosts static classification performance.

TABLE II
AVERAGE ACCURACY (%) ON MODELNET40 UNDER THE 3-WAY 5-SHOT FSCIL SETTING (I.E., NEW TASK $\mathcal{D}^t$ ($t > 1$) CONSISTS OF THREE CLASSES AND PER CLASS HAS FIVE TRAINING SAMPLES)

Method	Tasks									Our Relative Improvements
	1	2	3	4	5	6	7	8	9	
Fine-tune	96.41	93.42	82.68	62.21	54.35	38.84	37.55	28.44	18.32	+38.82
Joint train	96.41	95.82	82.01	75.15	65.21	62.94	58.56	56.93	52.74	+ 4.40
iCaRL [37]	95.63	87.13	78.16	67.61	45.24	30.09	29.62	29.01	26.29	+30.85
EEIL [49]	95.63	88.11	77.56	67.84	44.82	29.05	28.43	27.85	27.13	+30.01
NCM [50]	95.63	90.40	66.28	49.65	41.15	35.92	28.04	28.22	22.95	+34.19
I3DOL [10]	96.50	84.02	74.87	64.61	59.09	51.29	48.28	43.17	38.50	+18.64
PAConv [51]	96.74	86.41	82.74	73.43	59.64	52.73	49.22	38.83	38.70	+18.44
SDC [47]	96.41	91.25	81.07	77.24	64.15	61.54	56.00	50.02	45.73	+11.41
IDLVQ [40]	96.50	89.22	83.26	76.32	65.95	63.15	58.56	57.00	52.75	+ 4.39
C-FSCIL [52]	96.50	86.32	82.55	78.15	71.68	64.59	60.57	56.26	54.10	+ 3.04
FACT [53]	97.18	94.62	85.06	76.32	69.50	64.98	60.86	56.59	55.02	+ 2.12
Ours	**97.86**	**97.00**	**87.68**	**81.17**	**74.79**	**68.68**	**64.95**	**62.30**	**57.14**	**0**

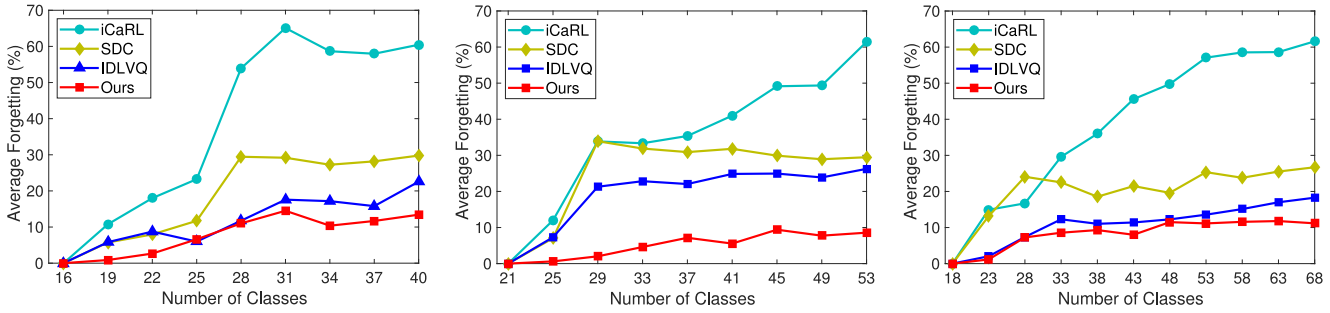

Fig. 5. Average forgetting of different methods on (a) ModelNet40, (b) ShapeNet, and (c) MCB datasets. Our method exceeds others by a large margin at the last task.

2) Our method significantly outperforms the fine-tune method by 38.82%, as it does not make any effort to handle catastrophic forgetting and overfitting.
3) Our method outperforms the joint train method (at least 4.40%) that suffers from overfitting due to many few-shot classes. In addition, the joint train method is time consuming and resource consuming as it uses all old training data.
4) Our method outperforms representative methods of CIL and FSCIL by about 2.12%–34.19%. The reason is that CIL and FSCIL methods cannot explore what 3-D topological characteristics are beneficial for mitigating catastrophic forgetting; however, our method can explore unique 3-D topological characteristics to mitigate catastrophic forgetting. In addition, our ESSA strategy can further mitigate catastrophic forgetting and overfitting.

Fig. 5(a) illustrates the forgetting curves of different methods across nine learning tasks on ModelNet40. We can observe that our method suffers from less forgetting than CIL methods and FSCIL method and obtains a 9.14% gain over the best exemplars-based method IDLVQ [40]. Combining Table II and Fig. 5(a), we can see that our method is more effective than other comparative methods on mitigating the catastrophic forgetting and overfitting of FS3DIL. We analyze the reasons as follows: Our embedding network abstracts adaptive and larger receptive field to construct discriminative local topological characteristics and focuses on unique 3-D topological characteristics to mitigate catastrophic forgetting; the model alignment regularization can mitigate catastrophic forgetting and overfitting simultaneously; the ESSA strategy can further mitigate the catastrophic forgetting and overfitting in the inference phase.

2) ShapeNet and MCB Results: The accuracy comparisons on ShapeNet and MCB datasets are reported in Tables III and IV, respectively. The forgetting curve comparisons are illustrated in Fig. 5(b) and (c). Some conclusions are drawn from these comparative experiments.

1) Comparing the accuracy of different methods at task 1, our method outperforms the others by at least 1.31% and 0.81% on ShapeNet and MCB datasets, respectively. These results demonstrate that the irregular meshes can be characterized well by our embedding network and meta-metric loss $\mathcal{L}_{\text{MML}}$, and our method can boost the static classification performance.
2) Our method significantly exceeds the fine-tune method and slightly exceeds the joint train method. Fine-tune method seriously suffers from catastrophic forgetting on old classes and overfitting on new classes. The joint train method suffers from overfitting and is time consuming and resource consuming.
3) In terms of accuracy, our method exceeds the SOTA IDLVQ method by 5.39% and 6.03% on ShapeNet and MCB datasets, respectively, and has significantly less forgetting by 17.68% and 7.11% on ShapeNet and MCB datasets, respectively. This is because these CIL and FSCIL methods cannot explore what 3-D topological characteristics are beneficial for mitigating catastrophic forgetting. Our method can explore unique

TABLE III
AVERAGE ACCURACY (%) ON SHAPENET UNDER THE 4-WAY 5-SHOT FSCIL SETTING

Method	Tasks									Our Relative Improvements
	1	2	3	4	5	6	7	8	9	
Fine-tune	84.28	50.33	31.98	23.82	16.09	15.17	13.42	10.29	8.86	+38.26
Joint train	84.28	73.48	59.19	53.95	52.47	49.41	45.55	44.21	42.05	+5.07
iCaRL [37]	83.63	72.66	47.91	48.41	39.94	33.40	22.44	22.09	15.96	+31.16
EEIL [49]	83.63	73.62	47.31	44.27	39.50	30.32	20.63	20.32	17.09	+30.03
NCM [50]	83.63	74.27	44.54	29.27	27.02	25.94	19.28	17.02	14.78	+32.34
SDC [47]	84.28	74.78	60.70	53.73	52.78	45.37	44.54	41.63	40.40	+6.72
IDLVQ [40]	84.32	74.48	61.02	54.20	53.54	49.94	44.74	43.32	41.43	+5.39
Ours	**85.63**	**76.12**	**68.16**	**65.27**	**60.79**	**56.29**	**52.5**	**50.27**	**47.12**	**0**

TABLE IV
AVERAGE ACCURACY (%) ON THE MCB DATASET UNDER THE 5-WAY 5-SHOT FSCIL SETTING

Method	Tasks											Our Relative Improvements
	1	2	3	4	5	6	7	8	9	10	11	
Fine-tune	95.25	41.72	24.92	22.05	18.93	16.07	12.93	9.57	7.96	7.42	5.42	+41.77
Joint train	95.25	81.38	67.26	62.39	57.8	56.72	56.39	49.52	50.55	45.92	42.49	+4.70
iCaRL [37]	94.53	71.87	61.93	47.56	37.48	27.66	25.34	17.56	13.92	12.12	10.10	+37.09
EEIL [49]	94.53	72.16	60.55	48.83	36.89	26.32	24.77	16.93	13.56	12.79	12.14	+35.05
NCM [50]	94.53	76.22	58.55	44.57	27.26	24.52	23.63	12.22	11.13	11.01	9.56	+36.63
SDC [47]	95.25	80.90	63.79	58.69	53.37	53.47	46.6	44.62	45.37	35.05	33.25	+13.94
IDLVQ [40]	95.57	82.36	69.01	63.47	59.70	58.00	57.97	51.70	49.26	45.03	41.16	+6.03
Ours	**96.38**	**84.57**	**72.99**	**66.75**	**62.56**	**61.8**	**58.22**	**54.82**	**52.06**	**50.05**	**47.19**	**0**

TABLE V
ACCURACY (%) ON THE CO3D DATASET UNDER 5-WAY 5-SHOT SETTING

Method	Tasks						Improve
	1	2	3	4	5	6	
Fine-tune	76.9	58.1	43.0	23.1	20.3	14.6	+41.7
Joint train	76.9	69.4	58.2	55.5	52.0	51.8	+ 4.5
iCaRL [37]	76.9	66.5	60.8	52.0	47.5	33.7	+22.6
EEIL [49]	76.9	67.1	61.7	53.4	49.2	35.5	+20.8
C-FSCIL [52]	78.0	63.0	58.8	56.2	53.5	52.0	+ 4.3
FACT [53]	77.3	70.0	62.7	60.4	56.9	54.3	+ 2.0
Ours	**78.8**	**68.7**	**63.2**	**61.8**	**56.8**	**56.3**	**0**

TABLE VI
TIME (TRAINING AND INFERENCE TIME) AND SPACE (NETWORK PARAMETERS) COMPLEXITY AT THE LAST TASK OF MODELNET40

Method	Training (ms)	Inference (ms)	#Param(M)
iCaRL [37]	6.9	1.4	0.377
SDC [47]	6.2	1.5	0.357
IDLVQ [40]	13.5	1.5	0.377
Ours	7.0	2.5	0.400

3-D topological characteristics to mitigate catastrophic forgetting. In addition, our ESSA further mitigates catastrophic forgetting and overfitting. In conclusion, our method not only outperforms others on static 3-D object classification but also outperforms others on FSI3DL.

3) CO3D Results: To compare the performance of different methods in the presence of noise, we present the accuracy comparisons of few-shot incremental learning on the CO3D dataset in Table V. For this dataset, we compute the overall accuracy of each task on all learned classes. We can observe that our method still suffers from less forgetting than other methods in the presence of noise and outperforms SOTA methods at least 2.0%. These experiments demonstrate that our method is more robust against noise than others.

4) Time and Space Complexity Analysis: We analyze time (training and inference time) and space (number of network parameters) complexity of our method and several representative methods at the last task of ModelNet40, the comparison results are depicted in Table VI. To compare the time complexity, we conduct time comparison experiments on the ModelNet40 dataset in the training and inference stage, respectively. For each 3-D object, the average training time of other comparative methods is 6.2–13.5 ms. The average training time of our method is 7.0 ms, which is faster than IDLVQ [40] and is comparable with iCaRL [37] and SDC [47]. For each 3-D object, the average inference time of other comparative methods is 1.4–1.5 ms. The average inference time of our method is 2.5 ms, which is comparable with comparative methods and is acceptable in practical application. To compare the space complexity, we statistical network parameters of our method and others. As depicted in Table VI, the parameters of our method are comparable with others.

D. Ablation Study

1) Contribution of Different Components: We conduct ablation studies on the ModelNet40 dataset to investigate the contribution of individual components on the performance of average accuracy. As depicted in Table VII, we analyze four variants of our method. 1) *E-EWC* [48]: we use the modified DGCNN as the embedding network that is optimized with $\mathcal{L}_{\text{EWC}}$ and triplet loss [57] as in SDC [47], and use NCM classifier without ESSF strategy as in SDC [47]; 2) *Ours-DGCNN*: embedding network replaced with DGCNN, and meta-metric loss replaced with triplet loss; 3) *Ours-Trip*: meta-metric loss replaced with triplet loss; and 4) *Ours-w/oTAGA*: without TAGA. The performance degrades 0.36%–21.16% when some components of our method are removed, which demonstrates that each designed component is indispensable in mitigating the catastrophic forgetting and overfitting of FS3DIL. The comparison between E-EWC and Ours-DGCNN demonstrates that our ESSF strategy can mitigate catastrophic forgetting and

TABLE VII
ABLATION STUDIES ON THE MODELNET40 DATASET FOR INVESTIGATING THE CONTRIBUTION OF INDIVIDUAL COMPONENTS ON AVERAGE ACCURACY (%)

Method	DGCNN	SVGC	TAGA	Triplet	MML	ESSF	Tasks								
							1	2	3	4	5	6	7	8	9
E-EWC	✓			✓			96.41	90.25	72.90	79.51	66.43	47.38	51.51	46.24	35.98
Ours-DGCNN	✓			✓		✓	96.41	95.06	79.09	80.16	68.27	62.45	62.96	54.74	53.00
Ours-Trip		✓	✓	✓		✓	97.38	95.82	85.65	79.78	73.19	68.08	63.59	59.54	55.33
Ours-w/oTAGA		✓			✓	✓	97.48	95.80	**88.55**	78.24	71.85	65.38	61.64	59.69	56.78
Ours		✓	✓		✓	✓	**97.86**	**97.00**	87.68	**81.17**	**74.79**	**68.68**	**64.95**	**62.30**	**57.14**

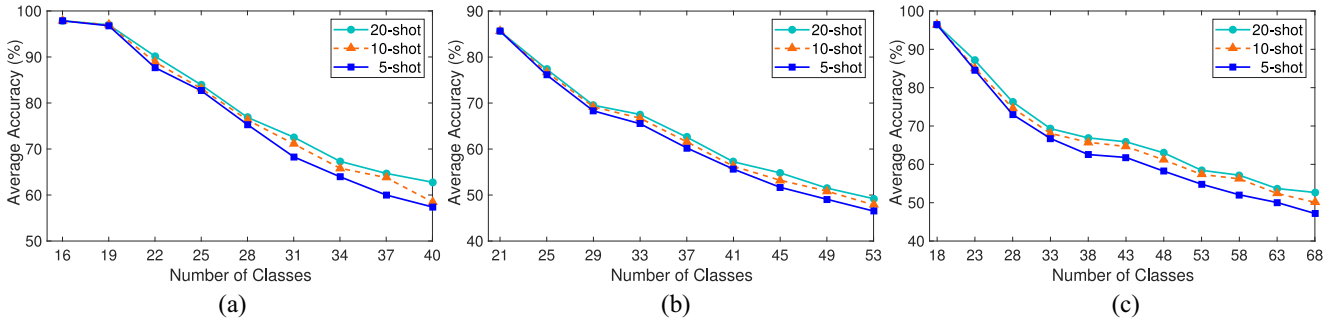

Fig. 6. Accuracy comparison of various FSCIL settings on (a) ModelNet40, (b) ShapeNet, and (c) MCB datasets.

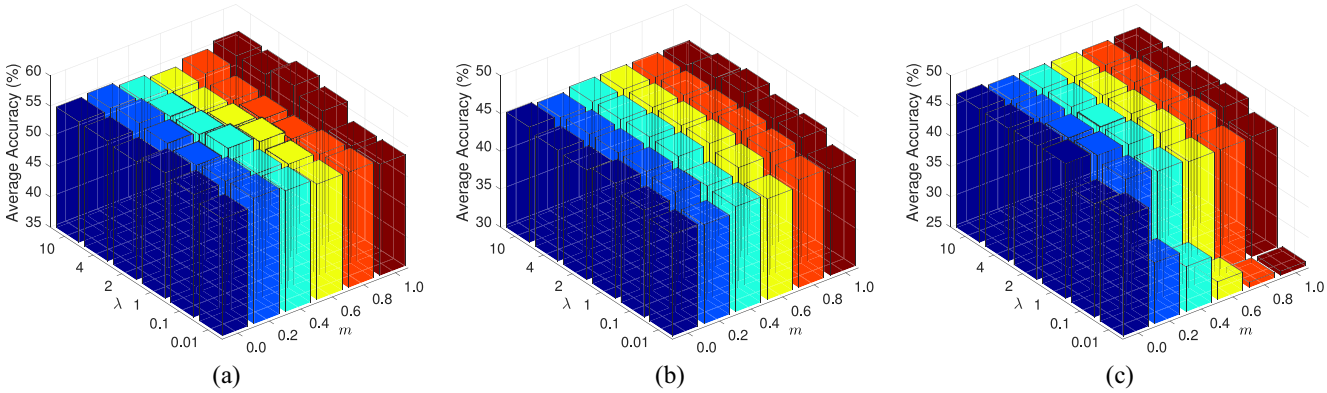

Fig. 7. Parameter investigations with respect to λ and m on (a) ModelNet40, (b) ShapeNet, and (c) MCB datasets. We vary regularization parameter λ in $[0.01, 0.1, 1, 2, 4, 10]$ times of 10^8.

overfitting and achieves a performance boost of 17.02%. The comparison between Ours-DGCNN and Ours-Trip demonstrates that our embedding network can extract more discriminative features and focus on unique topological characteristics than DGCNN and boosts the performance by 2.33%. The comparison between Ours-w/oTAGA and Ours further demonstrates that our TAGA module can explore beneficial 3-D topological characteristics to mitigate catastrophic forgetting and achieves an obvious performance boost at each task. The comparison between Ours-Trip and ours demonstrates that the meta-metric loss can better distinguish different classes with a gain of 1.81%.

2) Effect of Different Shot of Training Data: To investigate the effect of different numbers of training samples, we report the accuracy of our method under 5-shot, 10-shot, and 20-shot settings on three datasets, respectively. Fig. 6 illustrates that the performance of our method increases as increasing training samples, which shows the overfitting problem can be mitigated by increasing training samples.

3) Parameter Investigation: We evaluate the effect of changing regularization parameter λ and margin parameter m on the performance of our method over ModelNet40, ShapeNet, and MCB datasets. We vary regularization parameter λ in $[0.01, 0.1, 1, 2, 4, 10]$ times of 10^8 and vary margin parameter m in $[0.0, 0.2, 0.4, 0.6, 0.8, 1.0]$. From the illustrated results in Fig. 7, we can achieve the best performance on the ModelNet40 dataset with $\lambda = 2 * 10^8$ and $m = 1.0$, ShapeNet dataset with $\lambda = 2 * 10^8$ and $m = 1.0$ and MCB dataset with $\lambda = 10^8$ and $m = 0.0$. In addition, we evaluate the effect of changing threshold parameter τ of the ESSF strategy [i.e., (17)] on the ModelNet40 dataset. We vary threshold parameter τ in $[0.0, 1.0]$ with an interval of 0.1, as shown in Fig. 8(a). From the illustrated results, our method achieves the best performance on the ModelNet40 dataset when $\tau = 0.2$. The performance is constant when $\tau \geq 0.5$. This is because only $z_j = \omega_j^b \mathbf{z}_j^b + \omega_j^n \mathbf{z}_j^n$ is effective when $\tau \geq 0.5$.

We investigate the effect brought by the number of SVGC modules on ModelNet40 and MCB datasets, in other words, exploring the best embedding network structure. As illustrated in Fig. 8, we vary the number of SVGC from 1 to 3 and train each embedding network same epochs at each task (see Section IV-B). The results demonstrate that two SVGCs (i.e.,

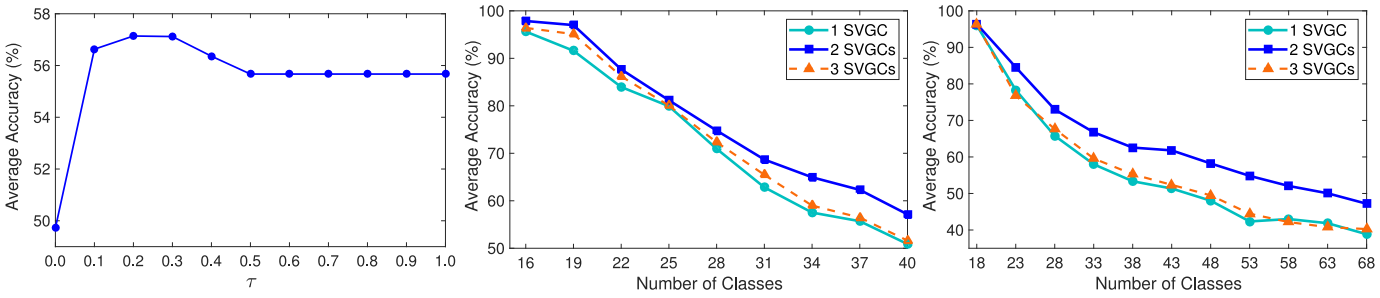

Fig. 8. (a) Parameter investigations threshold parameter τ. We vary τ in [0.0, 1.0] with an interval of 0.1. Our method achieves the best performance when $\tau = 0.2$. Comparisons of the various number of SVGC modules on (b) ModelNet40 and (c) MCB datasets. Two SVGCs achieve the best performance in each task and outperform others with a large margin.

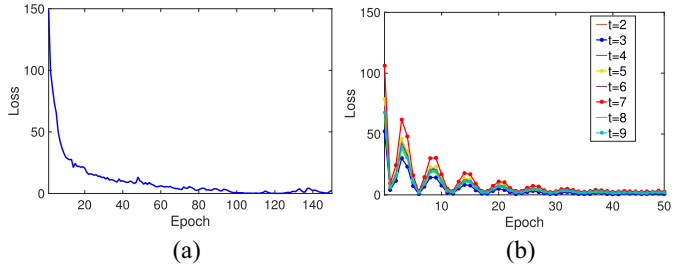

Fig. 9. Black Convergence analysis about all tasks on the ModelNet40 dataset. Our method starts to converge at about 100 epochs and 30 epochs for (a) task 1 (base task) and (b) task 2–9 (new tasks), respectively.

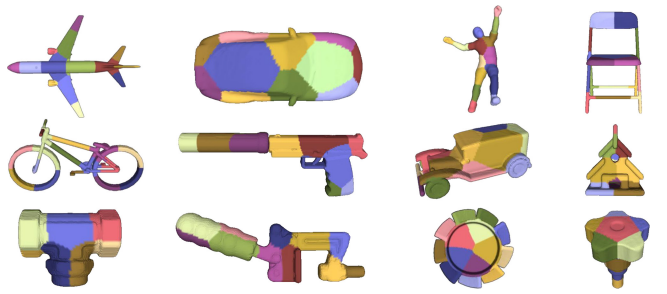

Fig. 10. Visualization of Laplacian spectral clustering results on ModelNet40 (first row), ShapeNet (second row), and MCB (third row) datasets.

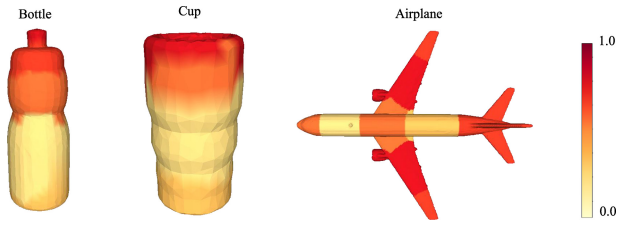

Fig. 11. Visualization results of TAGA's attention weights on three 3-D objects of the ModelNet40 dataset. The dark color represents larger attention weights (i.e., more unique 3-D topological structures) and vice versa. Such as the bottom part of bottle and cup, and the cylinder part of three 3-D objects are neglected common topological structures that are shown in light color.

our default network) achieve the best performance in each task and outperform others by a large margin. The network with one SVGC module is too shallow to learn much knowledge. However, the network with three SVGCs is too deep, which may overfit the few-shot new tasks.

4) Convergence Analysis: Fig. 9 investigates the convergence stability of our method on the ModelNet40 dataset. On the base task, our model starts to converge and presents stable performance when the iterative training epoch is about 100. On all new tasks, our model starts to converge when the iterative training epoch is about 30. Thus, we use 100 and 30 epochs for the base and new tasks, respectively.

5) Visualization of Clustering and TAGA Results: We visualize the Laplacian spectral clustering results of some 3-D meshes objects from three datasets in Fig. 10. These clustering results show that the Laplacian spectral clustering can capture a 3-D object's main local topological structures, which help the embedding network constructs discriminative local topological characteristics with adaptive and larger receptive field for representing the irregular 3-D meshes better.

As shown in Fig. 11, we visualize the attention weights of the TAGA module on three 3-D objects of the ModelNet40 dataset. TAGA outputs larger weights to stress unique 3-D topological structures and we visualize them with dark color. On the contrary, TAGA outputs small weights for common topological structures (e.g., the bottom part of bottle and cup; cylinder part of three 3-D objects) that are shown in light color and are neglected to mitigate catastrophic forgetting. It demonstrates our TAGA module can focus on unique 3-D topological characteristics while neglecting common topological characteristics to mitigate catastrophic forgetting.

6) Effect of Different Order of New Classes: To investigate the effect brought by different orders of new classes, we repeat the whole learning process ten times with different random seeds on the ModelNet40 dataset. The three 3-D object classification datasets are long-tailed distribution (or data imbalance), i.e., small portion of classes (head classes) have massive training samples but the other classes (tail classes) have only a few samples. According to the problem definition of FSI3DL, we choose head classes as base classes of the base/first task and choose the tail classes as few-shot classes of new tasks, which is a natural and reasonable. Thus, the classes of the first task are fixed in each line and only the classes of new tasks changed in each line. As shown in Table VIII, the experiments demonstrate different orders of new classes have some influences on the performance among

TABLE VIII
AVERAGE ACCURACY (%) OF DIFFERENT RANDOM SEEDS ON MODELNET40 UNDER THE 3-WAY 5-SHOT FSCIL SETTING (I.E., NEW TASK $\mathcal{D}^t$ ($t > 1$) CONSISTS OF THREE CLASSES AND FIVE TRAINING DATA PER CLASS). THE CLASSES OF THE FIRST TASK ARE FIXED IN EACH LINE, AND VARIOUS RANDOM SEEDS ARE ONLY USED FOR CLASSES NEW TASKS IN EACH LINE

Random seed	Tasks								
	1	2	3	4	5	6	7	8	9
1	97.86	96.97	90.7	85.06	76.97	69.77	65.64	63.89	57.32
2	97.86	96.68	85.14	81.12	72.10	66.48	63.83	61.55	57.25
3	97.86	96.69	84.74	78.60	75.70	66.42	63.17	60.66	57.00
4	97.86	96.90	90.74	83.14	75.01	70.71	64.60	61.36	57.24
5	97.86	96.70	87.75	81.67	76.58	69.86	64.71	61.92	57.03
6	97.86	96.68	89.63	79.25	71.69	70.37	63.75	60.58	57.09
7	97.86	97.28	91.13	82.79	75.92	67.62	64.55	62.75	57.14
8	97.86	97.28	85.26	80.41	75.93	69.52	66.7	63.76	57.16
9	97.86	97.60	85.04	76.48	74.72	69.20	65.57	62.81	57.18
10	97.86	97.22	86.67	83.18	73.28	66.85	66.98	63.72	56.99
Avg	**97.86**	**97.00**	**87.68**	**81.17**	**74.79**	**68.68**	**64.95**	**62.30**	**57.14**

TABLE IX
AVERAGE ACCURACY (%) OF DIFFERENT POOLING FUNCTIONS ON MODELNET40 UNDER THE 3-WAY 5-SHOT FSCIL SETTING

Pooling function	Tasks								
	1	2	3	4	5	6	7	8	9
sum	96.12	92.35	79.97	79.81	72.39	66.23	58.65	55.41	50.77
mean	96.50	90.64	83.03	79.02	73.02	66.82	60.24	57.91	53.53
max	**97.86**	**97.00**	**87.68**	**81.17**	**74.79**	**68.68**	**64.95**	**62.30**	**57.14**

the intermediate incremental tasks, but have a little influence on the average performance across all incremental tasks. Because our method could explore and accumulate useful 3-D topological characteristics of all previously learned classes to alleviate the catastrophic forgetting in the final task regardless of different classes orders.

In addition, we report the average test accuracy in the last row to further overcome the influence of different classes orders on the average performance across all incremental tasks. We also repeat the whole learning process ten times with different random seeds on ShapeNet and MCB datasets and report the average test accuracy in the last row of Tables III and IV.

7) Effect of Different Pooling Functions: We investigate the effect brought by the different pooling functions on ModelNet40 dataset, i.e., we replace the max-pooling function of SVGC module and the first GraphConv with avg-pooling function or sum-pooling function. As shown in Table IX, the experiments demonstrate different pooling functions have some significant influences on the performance among all tasks. The max-pooling outperforms avg-pooling and sum-pooling at each task by a large margin.

V. CONCLUSION

In this article, we propose a novel TopGCN that consumes 3-D meshes directly to address FSI3DL. Specifically, the SVGC module is designed to construct several main local topological characteristics for representing the irregular 3-D meshes better. Meanwhile, the TAGA module stresses unique 3-D topological characteristics that help mitigate catastrophic forgetting. We optimize the network by a meta-metric loss to better distinguish different classes. Moreover, we find the fine-tuning strategy with model alignment regularization can mitigate overfitting. Finally, an ESSF strategy is proposed in the inference phase to select proper embedding, which is capable of further mitigating overfitting and catastrophic forgetting. Extensive experiments on four 3-D datasets demonstrate the effectiveness of our method.

Compared with FSCIL methods in images, the FSI3DL is also very important since there are massive amounts of 3-D objects needed to manage in practical scenarios. We believe the proposed method can solve practical problems and bring extensive benefits. Compared with static 3-D object classification, the proposed FSI3DL is more practical as incrementally coming new classes with few data is very common. For example, some artificial intelligence tasks (e.g., robot manipulation and autonomous driving) need to percept the environment through 3-D information, where arising new object classes with few samples happen frequently. In the future, we plan to research few-shot incremental learning in 3-D object detection, which is a more challenging task than FSI3DL. The proposed method can be employed in the classification stage of few-shot incremental 3-D object detection since 3-D object classification is one subtask of 3-D object detection. Furthermore, during the inference stage, the FSI3DL model may encounter some unlearned classes and wrongly classifies them as learned classes. How to recognize these unlearned classes as unknown is also worth studying, which can make the FSI3DL more intelligent.

REFERENCES

[1] S. Yang, W. Wang, C. Liu, and W. Deng, "Scene understanding in deep learning-based end-to-end controllers for autonomous vehicles," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 1, pp. 53–63, Jan. 2019.

[2] J. Stria and V. Hlavác, "Classification of hanging garments using learned features extracted from 3D point clouds," in *Proc. IROS*, 2018, pp. 5307–5312.

[3] F. Hu et al., "Cyberphysical system with virtual reality for intelligent motion recognition and training," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 2, pp. 347–363, Feb. 2017.

[4] A. Sinha, J. Bai, and K. Ramani, "Deep learning 3D shape surfaces using geometry images," in *Proc. ECCV*, 2016, pp. 223–240.

[5] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. ICCV*, 2015, pp. 945–953.

[6] Z. Wu et al., "3D shapenets: A deep representation for volumetric shapes," in *Proc. CVPR*, 2015, pp. 1912–1920.

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. NeurIPS*, 2017, pp. 5099–5108.

[8] Y. Feng, Y. Feng, H. You, X. Zhao, and Y. Gao, "Meshnet: Mesh neural network for 3D shape representation," in *Proc. AAAI*, vol. 33, 2019, pp. 8279–8286.

[9] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends Cogn. Sci.*, vol. 3, no. 4, pp. 128–135, 1999.

[10] J. Dong, Y. Cong, G. Sun, B. Ma, and L. Wang, "I3DOL: Incremental 3D object learning without catastrophic forgetting," in *Proc. AAAI*, vol. 35, 2021, pp. 6066–6074.

[11] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or, "MeshCNN: A network with an edge," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–12, 2019.

[12] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong, "Few-shot class-incremental learning," in *Proc. CVPR*, 2020, pp. 12183–12192.

[13] A. Cheraghian, S. Rahman, P. Fang, S. K. Roy, L. Petersson, and M. Harandi, "Semantic-aware knowledge distillation for few-shot class-incremental learning," in *Proc. CVPR*, 2021, pp. 2534–2543.

[14] U. Von Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.

[15] H. Ren, Y. Cai, X. Chen, G. Wang, and Q. Li, "A two-phase prototypical network model for incremental few-shot relation classification," in *Proc. 28th Int. Conf. Comput. Linguist.*, 2020, pp. 1618–1629.

[16] A. X. Chang et al., "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*.

[17] S. Kim, H.-G. Chi, X. Hu, Q. Huang, and K. Ramani, "A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks," in *Proc. ECCV*, 2020, pp. 175–191.

[18] J. Reizenstein, R. Shapovalov, P. Henzler, L. Sbordone, P. Labatut, and D. Novotny, "Common objects in 3D: Large-scale learning and evaluation of real-life 3D category reconstruction," in *Proc. ICCV*, 2021, pp. 10901–10911.

[19] G. Chen, T. Zhang, J. Lu, and J. Zhou, "Deep meta metric learning," in *Proc. ICCV*, 2019, pp. 9547–9556.

[20] E. Ahmed et al., "A survey on deep learning advances on different 3D data representations," 2018, *arXiv:1808.01462*.

[21] Y. Cong, R. Chen, B. Ma, H. Liu, D. Hou, and C. Yang, "A comprehensive study of 3-D vision-based robot manipulation," *IEEE Trans. Cybern.*, vol. 53, no. 3, pp. 1682–1698, Mar. 2023.

[22] S. Li et al., "Cross-atlas convolution for parameterization invariant learning on textured mesh surface," in *Proc. CVPR*, 2019, pp. 6143–6152.

[23] A. Kanezaki, Y. Matsushita, and Y. Nishida, "Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints," in *Proc. CVPR*, 2018, pp. 5010–5019.

[24] D. Maturana and S. Scherer, "Voxnet: A 3D convolutional neural network for real-time object recognition," in *Proc. IROS*, 2015, pp. 922–928.

[25] D. Z. Wang and I. Posner, "Voting for voting in online point cloud object detection," *Robot. Sci. Syst.*, vol. 1, no. 3, pp. 10–15, 2015.

[26] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-CNN: Octree-based convolutional neural networks for 3D shape analysis," *ACM Trans. Graph.*, vol. 36, no. 4, p. 72, 2017.

[27] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. CVPR*, 2017, pp. 652–660.

[28] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *Acm Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.

[29] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on x-transformed points," in *Proc. NeurIPS*, 2018, pp. 820–830.

[30] Y.-L. Qiao, L. Gao, P. Rosin, Y.-K. Lai, and X. Chen, "Learning on 3D meshes with Laplacian encoding and pooling," *IEEE Trans. Vis. Comput. Graphics*, vol. 28, no. 2, pp. 1317–1327, Feb. 2022.

[31] B. Ma, Y. Cong, H. Liu, and X. Tang, "TPN: Topological perception network for 3D mesh representation," in *Proc. ICIP*, 2020, pp. 2711–2715.

[32] Y. Liu, B. Schiele, and Q. Sun, "An ensemble of epoch-wise empirical Bayes for few-shot learning," in *Proc. ECCV*, 2020, pp. 404–421.

[33] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. ICML*, 2017, pp. 1126–1135.

[34] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. NeurIPS*, vol. 29, 2016, pp. 3630–3638.

[35] C. Zhang, Y. Cai, G. Lin, and C. Shen, "DeepEMD: Few-shot image classification with differentiable earth mover's distance and structured classifiers," in *Proc. CVPR*, 2020, pp. 12203–12213.

[36] J. Dong et al., "Federated class-incremental learning," in *Proc. CVPR*, 2022, pp. 10164–10173.

[37] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," in *Proc. CVPR*, 2017, pp. 2001–2010.

[38] G. Sun, Y. Cong, Q. Wang, B. Zhong, and Y. Fu, "Representative task self-selection for flexible clustered lifelong learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 4, pp. 1467–1481, Apr. 2022.

[39] G. Sun, Y. Cong, J. Dong, Y. Liu, Z. Ding, and H. Yu, "What and how: Generalized lifelong spectral clustering via dual memory," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3895–3908, Jul. 2022.

[40] K. Chen and C.-G. Lee, "Incremental few-shot learning via vector quantization in deep embedded space," in *Proc. ICLR*, 2021.

[41] C. Zhang, N. Song, G. Lin, Y. Zheng, P. Pan, and Y. Xu, "Few-shot incremental learning with continually evolved classifiers," in *Proc. CVPR*, 2021, pp. 12455–12464.

[42] T. Martinetz and K. Schulten, "A "neural-gas" network learns topologies," in *Proc. Artif. Neural Netw.*, 1991, pp. 397–402.

[43] B. Fritzke, "A growing neural gas network learns topologies," *NeurIPS*, vol. 7, 1994, pp. 625–632.

[44] A. Sato and K. Yamada, "Generalized learning vector quantization," in *Proc. NeurIPS*, 1996, pp. 423–429.

[45] G. Li, M. Muller, A. Thabet, and B. Ghanem, "DeepGCNs: Can GCNs go as deep as CNNs?" in *Proc. ICCV*, 2019, pp. 9267–9276.

[46] R. Nagar and S. Raman, "Fast and accurate intrinsic symmetry detection," in *Proc. ECCV*, 2018, pp. 417–434.

[47] L. Yu et al., "Semantic drift compensation for class-incremental learning," in *Proc. CVPR*, 2020, pp. 6982–6991.

[48] J. Kirkpatrick et al., "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci.*, vol. 114, no. 13, pp. 3521–3526, 2017.

[49] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, "End-to-end incremental learning," in *Proc. ECCV*, 2018, pp. 233–248.

[50] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *Proc. CVPR*, 2019, pp. 831–839.

[51] M. Xu, R. Ding, H. Zhao, and X. Qi, "PAConv: Position adaptive convolution with dynamic kernel assembling on point clouds," in *Proc. CVPR*, 2021, pp. 3173–3182.

[52] M. Hersche, G. Karunaratne, G. Cherubini, L. Benini, A. Sebastian, and A. Rahimi, "Constrained few-shot class-incremental learning," in *Proc. CVPR*, 2022, pp. 9057–9067.

[53] D.-W. Zhou, F.-Y. Wang, H.-J. Ye, L. Ma, S. Pu, and D.-C. Zhan, "Forward compatible few-shot class-incremental learning," in *Proc. CVPR*, 2022, pp. 9046–9056.

[54] J. Huang, H. Su, and L. Guibas, "Robust watertight manifold surface generation method for shapenet models," 2018, *arXiv:1802.01698*.

[55] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proc. 24th Annu. Conf. Comput. Graph. Interact. Techn.*, 1997, pp. 209–216.

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[57] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *Proc. Int. Workshop Simil. Based Pattern Recognit.*, 2015, pp. 84–92.

[58] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *Proc. CVPR*, 2017, pp. 3366–3375.

[59] M. Masana, X. Liu, B. Twardowski, M. Menta, A. D. Bagdanov, and J. van de Weijer, "Class-incremental learning: Survey and performance evaluation on image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5513–5533, May 2013.

[60] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proc. ECCV*, 2018, pp. 532–547.

Bingtao Ma received the B.S. degree from the Hebei University of Technology, Tianjin, China, in 2017. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Robotics, Shenyang Institute of Automation, University of Chinese Academy of Sciences, Shenyang, China.

His current research interests include 3-D computer vision, transfer learning, and continual learning.

Yang Cong (Senior Member, IEEE) received the B.Sc. degree from Northeast University, Shenyang, China, in 2004, and the Ph.D. degree from the State Key Laboratory of Robotics, Chinese Academy of Sciences, Shenyang, in 2009.

He is a Full Professor with the Chinese Academy of Sciences. He was a Research Fellow with the National University of Singapore, Singapore, and Nanyang Technological University, Singapore, from 2009 to 2011, respectively; and a Visiting Scholar with the University of Rochester, Rochester, NY, USA. He has authored over 70 technical papers. His current research interests include image processing, compute vision, machine learning, multimedia, medical imaging, data mining, and robot navigation.

Prof. Cong won the National Science Fund for Excellent Young Scholars (NSFC), the National Natural Science Foundation of China for Distinguished Youth Fund (NSFC), the First Prize of Natural Science Award of Liaoning Province, and the First Prize of Natural Science Award of Chinese Association of Automation. He has served on the editorial board of the *Journal of Multimedia*. He also serves as an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and other journals.

Jiahua Dong received the B.S. degree from Jilin University, Changchun, China, in 2017. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Robotics, Shenyang Institute of Automation, University of Chinese Academy of Sciences, Shenyang, China.

He also has some top-tier conference papers accepted at CVPR, ICCV, ECCV, and AAAI. His current research interests include transfer learning, robotic vision, and medical image processing.