sensors

MDPI

Article

Progressive Deep Learning Framework for Recognizing 3D Orientations and Object Class Based on Point Cloud Representation †

Sukhan Lee [1,2,*] and Yongjun Yang [2]

[1] Artificial Intelligence Department, Sungkyunkwan University, Suwon 16419, Korea
[2] School of Information & Communication Engineering, Sungkyunkwan University, Suwon 16419, Korea; yongjun823@skku.edu
* Correspondence: lsh1@skku.edu; Tel.: +82-(31)-299-6471

† This work is an extension of our conference paper "Progressive Framework of Learning 3D Object Classes and Orientations from Deep Point Cloud Representation", IEEE IMCOM 2020, Taichung, Taiwan, 3–5 January 2020.

Abstract: Deep learning approaches to estimating full 3D orientations of objects, in addition to object classes, are limited in their accuracies, due to the difficulty in learning the continuous nature of three-axis orientation variations by regression or classification with sufficient generalization. This paper presents a novel progressive deep learning framework, herein referred to as 3D POCO Net, that offers high accuracy in estimating orientations about three rotational axes yet with efficiency in network complexity. The proposed 3D POCO Net is configured, using four PointNet-based networks for independently representing the object class and three individual axes of rotations. The four independent networks are linked by in-between association subnetworks that are trained to progressively map the global features learned by individual networks one after another for fine-tuning the independent networks. In 3D POCO Net, high accuracy is achieved by combining a high precision classification based on a large number of orientation classes with a regression based on a weighted sum of classification outputs, while high efficiency is maintained by a progressive framework by which a large number of orientation classes are grouped into independent networks linked by association subnetworks. We implemented 3D POCO Net for full three-axis orientation variations and trained it with about 146 million orientation variations augmented from the ModelNet10 dataset. The testing results show that we can achieve an orientation regression error of about 2.5° with about 90% accuracy in object classification for general three-axis orientation estimation and object classification. Furthermore, we demonstrate that a pre-trained 3D POCO Net can serve as an orientation representation platform based on which orientations as well as object classes of partial point clouds from occluded objects are learned in the form of transfer learning.

Keywords: orientation representation; 3D point cloud; 3D object; progressive learning; association network

Citation: Lee, S.; Yang, Y. Progressive Deep Learning Framework for Recognizing 3D Orientations and Object Class Based on Point Cloud Representation. *Sensors* **2021**, *21*, 6108. https://doi.org/10.3390/s21186108

Academic Editor: Francesco Bellotti

Received: 5 July 2021
Accepted: 8 September 2021
Published: 12 September 2021

1. Introduction

Data representation is crucial when dealing with 3D objects. As far as data representation for 3D objects is concerned, there are three approaches available currently: (1) multiple 2D images from different perspectives, (2) voxel or octree representation and (3) 3D point cloud or mesh representation. Among them, 3D point cloud representation presents the most efficient means of representing 3D objects, featured with order-independence in its data structure. In addition, 3D point cloud representation is further supported by the availability of low-cost yet highly robust real-time RGB-D cameras. More significantly, the recent advancement of deep point networks, such as PointNet [1], FoldingNet [2] and their variants [3], demonstrates that 3D point clouds can be effectively processed by deep point networks for classification, segmentation and reconstruction with high accuracy

and generalization power. In general, deep point networks employ a point-wise multi-layer mapping approach with shared weights, while choosing the maximum point values from individual axes of the mapped space to define the global features as a means of exploiting order-independence. Moreover, deep point networks can be trained based on a number of publicly available 3D object datasets, including ModelNet [4], PASCAL3D+ [5], ShapeNet [6], LineMod [7] and OCID [8], where 3D point clouds are either from object CAD models or from actual measurements. The deep point networks trained with 3D point cloud representation of 3D objects can effectively serve as a platform for the classification of 3D objects. Notably, however, as far as 3D objects are concerned, it is not only their classes, but also their poses, i.e., their orientations, that are important attributes to be represented. However, unlike object classes, 3D object orientations represent continuous variations about three independent rotational axes that pose a challenge in precisely identifying orientations using deep learning–based classification or regression.

To date, deep learning approaches for 3D object recognition and orientation estimation are focused on relaxing limitations through a trade-off between precision and complexity. To deal with the trade-off, hybrid approaches are introduced such that the strength of deep learning approaches for object detection and recognition and that of conventional vision technologies for high precision orientation estimation are combined [9]. In other words, hybrid approaches seek for precision in orientation estimation at the expense of computational cost associated with conventional vision technologies. Should end-to-end deep learning approaches to orientation estimation be considered, they have to limit the number of orientation classes or the precision in regression to a manageable level [10]. Recently, a number of end-to-end deep learning approaches for 6D object pose estimation based on RGB and RGB-D data have been proposed. They show that end-to-end deep learning approaches can possibly achieve a sufficient level of accuracy in pose estimation, while taking full advantage of the processing speed provided by deep networks. They solve the problem of the precision–complexity trade-off by combining feature-induced regression, using local and global RGB or RGB-D features with deep iterative 6D pose refinement, supported by a powerful semantic segmentation of objects from a scene.

Despite the recent progress, the precision–complexity trade-off remains a fundamental issue for deep learning approaches in 6D pose estimation. In fact, this trade-off represents a general problem common to the estimation of multi-variate continuous functions through either classification or regression. Furthermore, should we include additional variations, such as occlusions, to represent 3D objects, the trade-off becomes even worse, due to the increased complexity. In this paper, we propose to solve the precision–complexity issue based on a progressive framework for learning the object class and three axes of orientation variations. The proposed framework is configured with four independent networks representing object class and three axes of orientations that are connected by in-between feature association subnetworks. The proposed framework progressively learns the object class and three axes of orientation variations with training samples limited only to pertinent variations. Instead, in-between feature association subnetworks learn to cover full data representations for independent networks. With the proposed progressive framework, we intend to develop a deep learning network that serves as a representation platform for 3D objects based on point cloud representation of 3D objects. As a platform, the proposed network is expected to be easily extended to learn partial point cloud representation of 3D objects, due to occlusion.

1.1. Related Work

Currently, the approaches proposed for the representation of 3D objects include the following: (1) multiple 2D images from different perspectives [11–14], (2) voxel representation and its variants, such as a hybrid grid–octree data structure [4,11,15–18], (3) 3D point cloud representation [19,20] and (4) mesh representation [21]. For deep learning approaches, voxel representation allows direct extension of the methodologies that are well established for 2D convolutional neural networks (CNNs). However, direct application of

CNNs to voxel representation of 3D objects may suffer from computational cost, due to the inclusion of a large number of voxels with no actual contribution, although multi-level octree representation can reduce such computational cost to a certain degree [21]. On the other hand, point cloud representation is more efficient but requires special order independent processing, such as the one done by PointNet and FoldingNet, where the global geometric features are extracted based on a point-wise mapping with shared weights and a max selection operation. Note also the recent emergence of approaches to reinforce 3D object representation with structural or topological information based on graph or mesh convolutional networks [20–22].

Traditionally, 6D pose estimation is done by matching the extracted features with those of the ground truth object models [23]. Such traditional vision approaches may offer high precision in pose estimation, provided that a sufficient number of features can be extracted and matched for pose estimation. However, they suffer from high computational cost as a result of extracting and matching hand-crafted photometric and geometric features, besides the lack of robustness and generalization in dealing with variations. The recent advancement in deep learning networks offers an opportunity to overcome the limitation of traditional approaches by presenting a powerful platform for the detection, recognition and segmentation of objects in a cluttered scene [24–26]. Deep learning–based object detection, recognition and segmentation platforms are able to provide not only robustness and generalization in performance, but also fast processing speeds. The availability of such deep learning platforms enables the development of hybrid approaches [27], where deep learning approaches for object detection, recognition and segmentation are combined with traditional approaches to pose estimation so as to achieve high accuracy and speed at a reduced cost. Recently, end-to-end deep learning approaches for 6D pose estimation have emerged, where not only object detection, recognition and segmentation, but also 6D pose estimation are conducted by deep learning networks so that both accuracy and speed are at their maximum. For instance, RGB-based approaches determine 6D object poses either by directly regressing a quaternion representation of object orientations [28], by iteratively matching the rendered object view against the captured object image [12], or by predicting 3D coordinates of each object pixel through an auto-encoder generator, using a GAN framework. This is followed by iterative computation of the PnP algorithm with RANSAC [12]. In addition, RGB-D–based end-to-end deep learning network approaches are proposed for 6D pose estimation, in which pixel-wise embedding of color and point cloud, as well as the global feature representing both embedding, are generated and concatenated to regress pixel-wise 6D pose with the refinement of residual pose errors [29]. Lastly, it is worthwhile to introduce approaches extending pose estimation from an instance level to a category level, for instance, based on a pose-aware image generator trained by VAE for iterative optimization of object pose and shape [30], and a canonical representation of object categories with deep networks for estimating the object pose and size [31].

Recently, a progressive deep learning framework was proposed as a means of exploring the ability to transfer knowledge learned from prior tasks to a new task via lateral connections [32]. Such a progressive framework can be effective for learning multiple tasks or a complex task configured with multiple subtasks by correlating their embedded structure of data or knowledge. For instance, progressive frameworks have been applied to learning a variety of games in complex reinforcement learning domains [33] as well as modeling the acoustic features of noisy speech based on the knowledge transfer between different noise conditions [34]. Alternatively, progressive frameworks are adopted to recognize images having different visual complexities based on a set of network units activated sequentially with progressively increasing complexities, or to transfer knowledge between three paralinguistic tasks: speaker, emotion, and gender recognition, by exploiting how knowledge captured in one emotion dataset can be transferred to another [35]. Progressive deep learning frameworks have been reported to offer efficiency in learning with faster convergence and improvement in performance over conventional pre-training and fine-tuning with transfer learning [36].

1.2. Problem Statement and Proposed Approach

The fundamental issue to address here is how precise and accurate a deep learning network could be for estimating or predicting general three-axis orientations of 3D objects. This represents a general problem associated with deep learning approaches regarding their capability for approximating continuous functions with high dimensional input–output relationships. Recently, deep learning approaches to general three-axis orientation or 6D pose estimation of 3D objects based on classification and regression have shown considerable progress in their precision and accuracy toward the level offered by conventional hand-crafted feature engineering approaches [12,29,37]. However, further improvement in precision and accuracy, possibly to the level required by object manipulation tasks in various industrial applications, remains a necessity. Such improvement makes deep learning approaches highly preferable to conventional approaches for a wide range of applications with the advantages in robustness, generalization and computational speed. To tackle the above issue, we pay attention to how different ways to structure deep learning networks for estimating general three-axis orientations affect the performance in precision and accuracy. To be more specific, precision in orientation estimation relies on the number of classes to output, whereas accuracy in classification depends on the degree of data variations that individual output classes should generalize as well as the number of training data available for individual output classes. For higher precision and accuracy, we prefer defining a larger number of output classes, a smaller degree of data variations to generalize by each output class, and a larger number of training data available. On the other hand, for higher efficiency, we prefer a smaller number of output classes for structural simplicity manageable by available training data and computational power. As such, the structure of a deep learning network for orientation classification should be optimized in terms of the number of output classes, data variations associated with individual output classes, training data available as well as structural simplicity under optimal trade-off among precision, accuracy and efficiency. In particular, we need to address the issue caused by the exponential growth in the number of classes, as the orientation resolution represented by individual classes is increased for high precision estimation. Note that the above observation on precision and accuracy in orientation estimation in conjunction with classification structure can equally be applied to regression. This is because, in principle, they rely on the same structure for building embedding before outputs are trained in terms of regression or classification [37]. In fact, in this paper, we present both classification and regression for orientation estimation, where regression outputs are obtained as a weighted sum of class outputs with the weights given by class probabilities. Note that it is also possible to build a fully connected network on top of classification outputs to train for continuous orientation estimation, instead of a weighted sum of classification outputs with the weights from the probability distribution of output classes. We conjecture that classification-based regression has an advantage in that classification outputs play a role as control points in fitting data into a continuous regression function, where the distances of the data to control points may be used for local refinement of the regression function.

The structure of conventional deep learning approaches to orientation classification can be categorized into “fanned”, “grouped” and “hierarchical” (Figure 1).

By “fanned”, we mean that all individual three-axis orientation classes are separately represented as individual output classes [10,19,37]. By “grouped”, we mean that all individual three-axis orientation classes are clustered into three separate groups: x-, y-, and z-axis orientation class groups, where the x-axis group consists of only x-axis orientation classes with all y- and z-axis orientations clustered into x-axis orientation classes, and so on. By “hierarchical”, we mean that objects are classified hierarchically, e.g., in a hierarchy of x-, y- and z-axis classifications [9]. Table 1 shows the number of output classes, the data variations associated with individual output classes, the training data available for individual output classes as well as the simplicity in network structures associated with the above three conventional structures.

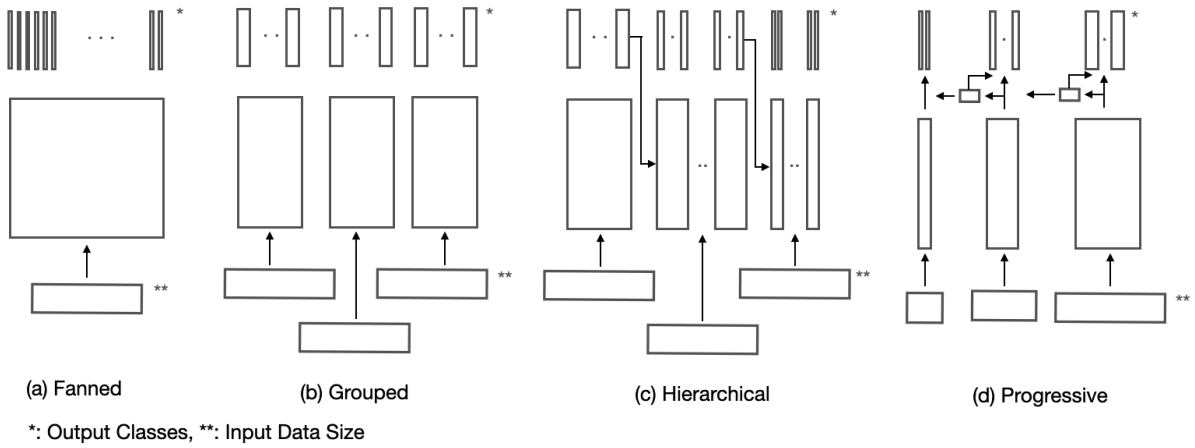

Figure 1. Feasible deep learning structures for estimating three-axis orientation estimation based on classification, including the proposed progressive structure.

Table 1. Performance comparison of the proposed progressive structure with other typical structures for classification in terms of the number of output classes, degree of data variations and the number of training data per output class.

Structure	No. of Output Class	Degree of Data Variation per Output Class	No. of Training Data per Output Class
Fanned	N == M^3	1	D/N(==M^3)
Grouped	3M	3	D/M
Hierarchical	N + M^2 + M	3, 2, 1 for 1st, 2nd, 3rd Layer	D/M, D/M^2, D/M3 for 1st, 2nd, 3rd Layer
Progressive	3M	1, 2, 3 for 1st, 2nd, 3rd Process	D/M3, D/M2, D/M for 1st, 2nd, 3rd Process

N: Total no. of classes(N == M3), M: no. of classes for each axis, D: Total no. of training data.

Generally speaking, for classification with a small number of orientation classes, a fanned architecture may be adopted. However, for classification with a large number of orientation classes, a grouped structure may be preferred for network simplicity at the expense of some accuracy. In between, we may consider a hierarchical structure as a compromise.

In this paper, we propose a "progressive" structure for deep learning–based orientation classification, as an alternative to the above conventional structures, that can handle a large number of orientation classes by a small number of output classes, yet with a reduced degree of data variations. The proposed progressive structure achieves this by progressively learning the embedding of x-, y- and z-axis orientation classes one after another and, at the same time, by progressively extending the degree of data variations associated with individual axes and learning association between the embedding of two-axis orientation classes in sequence along the progression. For example, as illustrated in Figure 1, first, one axis class outputs are trained to learn their embedding with other axes data variations fixed to their reference orientations. Then, another axis class outputs are trained to learn their embedding with the remaining axis data variations fixed as their reference orientation and, at the same time, to learn the association between the current and the prior axis embedding. Finally, the last axis class outputs are trained to learn their embedding while learning the association between the current and the prior axis embedding. Table 1 shows the comparison of the proposed progressive structure with the conventional ones in terms of the number of output classes, the data variations associated with individual output classes, the training data available for individual output classes as well as the simplicity in network structures. It indicates that the proposed progressive structure allows the number of output classes to be the same as that of a grouped structure, yet allows the degree of data variations to be the same as that of a hierarchical structure in such a way that high

precision and accuracy in the orientation estimation, closer to a fanned structure, can be achieved with a simple network structure.

2. D Point Cloud Based Object Class and Orientation Estimation Network: 3D POCO Net

As described in Section 1.2, we provide a deep learning network as a platform for representing object class and orientations in high precision and accuracy based on point cloud representation of 3D objects. To this end, a progressive framework for learning three axes of the orientation variations as well as the object class is designed and implemented based on independent networks that are connected by in-between feature association subnetworks (Figure 2). The proposed framework progressively learns the object class and three axes of orientation variations with training samples limited only to pertinent variations. Instead, in-between feature association subnetworks learn to cover full data representations for independent networks. The framework is based on the order-independent point cloud representation of PointNet for simplicity in implementation and computational efficiency. The progressive framework of networks thus implemented for object classification and orientation estimation is referred to here as the 3D Point Cloud Based Object Classification and Orientation Estimation Network or 3D POCO Net, in short form. 3D POCO Net is composed of the reference network, which outputs the classes associated with the 3D objects in their reference orientations and the three independent orientation networks, which generate their own orientations representing three consecutive rotations from the reference orientation. The reference network and the three orientation networks are linked by the association subnetworks that are trained to output the global features learned by the adjacent networks that they are linked to. Then, orientations of individual axes are estimated based on the weighted sum of orientation classes with the weight given as the class probabilities generated by the respective orientation networks. In the subsequent sections, we present the details of the network configuration and training procedure.

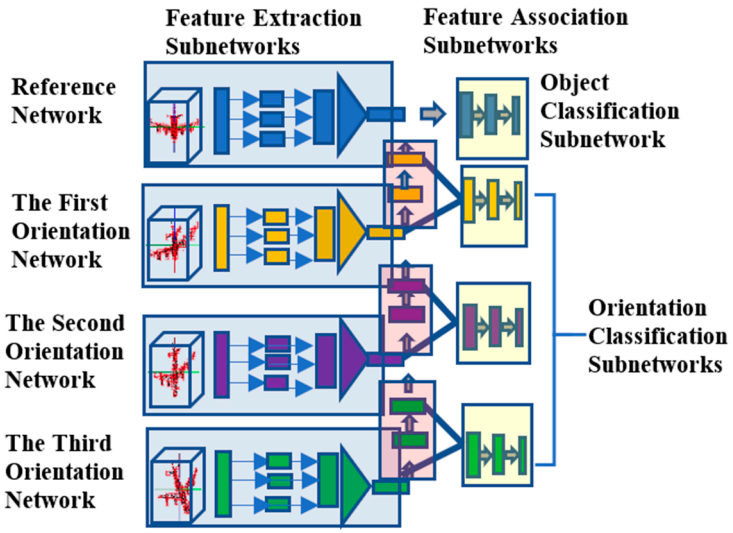

Figure 2. The proposed progressive framework with reference, first, second and third orientation networks.

2.1. Network Configuration

Reference Network: The reference network is composed of the "feature extraction subnetwork" and "object classification subnetwork" (Figure 2). The aim of the feature extraction subnetwork is to extract the global features associated with the 3D objects in their reference orientations based on PointNet with no T-Net for orientation compensation. The feature extraction subnetwork is configured with five weight-sharing hidden layers in [64, 64, 64, 128, 1204] format for individual 3D point, where each layer is followed by non-linear activation and max-pooling operations. A 1024 dimension of the global feature vectors is then extracted by applying the element-wise max selection operation to the output of

the last hidden layer. The resultant global feature vectors are then fed to the three-layered fully connected network of [512, 256, n_c] configuration, with n_c representing the number of object classes. This object classification subnetwork adopts the Leaky ReLU activation and batch normalization [38] for all the layers, except for the last decision layer. In addition, a dropout layer with a dropout rate of 0.3 is used just before the decision layer. The Adam optimizer is used to optimize the network parameters.

Orientation Networks: Three orientation networks are configured in 3D POCO Net to generate their particular axes of rotations, such as roll, pitch and yaw angles. Each orientation network is composed of the "feature extraction subnetwork", "orientation classification subnetwork" and "feature association subnetwork". Apart from the reference network, each orientation network has the feature association subnetwork that is trained to generate the global features of the adjacent network. The feature association subnetwork is configured as a stack of fully connected layers of [1024, 768, 512, 768, 1024]. The orientation classification subnetwork concatenates the two outputs from its feature association and feature extraction subnetworks for use as input in order to obtain the probability of the predefined number of orientation classes as the output. The orientation classification subnetwork is configured as a stack of fully connected layers of [2048, 1024, 512, 256, n_o], where no represents the number of predefined orientation classes. The orientation classification subnetwork adopts the Leaky ReLU activation and batch normalization for all the layers, except the last decision layer. Similarly, a dropout layer with a dropout rate of 0.3 is used just before the decision layer.

2.2. Training Procedure

The training of 3D POCO Net starts with training of the reference network for object classification based on the reference samples, i.e., the 3D object sub-dataset with the reference orientation as the input. Then, the trained reference network is kept fixed as the first orientation network is trained to learn its feature association and orientation classification subnetworks based on the first orientation samples, i.e., the 3D object sub-dataset generated by the first axis of rotation of the reference samples (Figure 3). In particular, this is followed by retraining the object classification subnetwork of the reference network in order to fine-tune based on the additional training samples available from the initial orientation samples. The same training procedure is repeated as the training proceeds to individual networks in the order of their association, except that retraining is applied to all the prior classification subnetworks one by one in the reverse order. The details of the training procedure are given in the following steps:

Step 1: The feature extraction and the object classification subnetworks of the reference network are trained using the reference sample data by optimizing the following loss function:

$$L_r(\{y_i\}, \{u_i\}) = \sum_i L_{cls}(f_r(r(y_i)), u_i) \quad (1)$$

where y_i and u_i represent the i^{th} reference input sample and its object class, respectively; $r(y_i)$ represents the extracted global feature of y_i from the feature extraction subnetwork; $f_r(r(y_i))$ is the output of the object classification subnetwork; and L_{cls} denotes SoftMax log classification loss.

Step 2: Once the training of the reference network is completed, the first orientation network is then trained, while the reference network trained is fixed, with the following objectives: 1) to make the output of its feature association subnetwork, $r'(x_{i1})$, x_{i1} be the i^{th} first orientation sample, which is equal to the output of the feature extraction subnetwork, $r(y_i)$, of the reference network; 2) to have its orientation classification subnetwork output, $f_o(r'(x_{i1}), p(x_{i1}))$, same as the true orientation class, a_{i1}, where the input of the orientation classification subnetwork is obtained by concatenating the output of its feature association subnetwork, $r'(x_{i1})$, and the output of its feature extraction subnetwork, $p(x_{i1})$; and 3) to ensure that the first orientation sample, x_{i1}, also satisfies its object class constraint, i.e., $f_r(r'(x_{i1}))$ is equal to u_i. The feature association subnetwork, the orientation classifica-

tion subnetwork and the feature extraction subnetwork of the orientation network are simultaneously trained by optimizing the following overall loss function:

$$L_o(\{x_{i1}\},\ \{y_i\},\{u_i\},\{a_{i1}\}) = \sum_i L_{cls}(f_o(r'(x_{i1}),p(x_{i1})),a_{i1}) + \beta * \sum_i \sqrt{||r'(x_{i1}) - r(y_i)||_2} + (1-\beta) * \sum_i L_{cls}(f_r(r'(x_{i1})),u_i) \quad (2)$$

where β represents the weight that balances the contributions of the feature association error and the object classification error. The loss is minimized based on the stochastic gradient decent optimization.

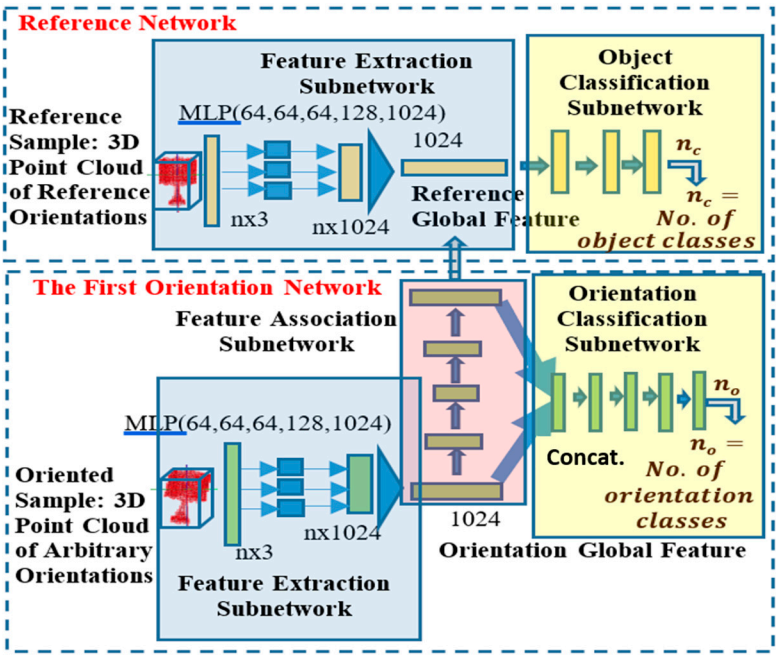

Figure 3. Illustration of the detailed configuration of 3D POCO Net with reference and first orientation networks.

Step 3: The refinement of the object classification subnetwork of the reference network is then followed based on the first orientation sample data available from Step 2, while all the parameters of other subnetworks are kept fixed. Specifically, the dataset for refining the object classification subnetwork is created by randomly mixing all the pair-wise data, $\{r(y_i),\ u_i\}$ and $\{r'(x_{j1}),\ u_j\}$, representing the global features and their object classes of the reference samples and the first orientation samples, respectively, in order to form $\{g_k,\ u_k\}$= $\{r(y_i),\ u_i\}\ U\{r'(x_{j1}),u_j\ \}$, k = either i or j. The object classification subnetwork of the reference network is then retrained by optimizing the following loss function:

$$L_o(\{g_k,\ u_k\}\ ) = \sum_k L_{cls}(f_r(g_k),u_k) \quad (3)$$

Step 4: Once the training of the first orientation network is completed, the second orientation network is trained in the same way as the first orientation network, and the second orientation sample dataset is generated by rotating the first orientation sample dataset about the second axis of rotation. First, with the first orientation network and the already trained reference network being kept fixed, the feature association subnetwork and the orientation classification subnetwork of the second orientation network are trained first by making their outputs, $r'(x_{i2})$ and $f_o(r'(x_{i2}),p(x_{i2}))$, become equal to the output, $p(x_{i1})$, of the feature extraction subnetwork of the first orientation network and the true second orientation class, a_{i2}, respectively. However, the training of the second orientation network

should ensure that not only the second orientation sample, x_{i2}, satisfies its first orientation class, i.e., $f_o(r'(\ x_{i1} = r'(x_{i2}))\ r'(x_{i2})) = a_{i1}$, but also that it satisfies its object class, i.e., $f_r(r'(\ x_{i1} = r'\ (x_{i2}))) = u_i$. Therefore, the training of the second orientation network is based on the following loss function:

$$\begin{aligned} L_o(\{x_{i2}\},\ \{x_{i1}\},\ \{a_{i2}\},\ \{a_{i1}\},\ \{y_i\},\{u_i\}) = \sum_i L_{cls}\left(f_o\left(r'^{(x_{i2})}, p(x_{i2})\right), a_{i2}\right) + \\ \alpha * \sum_i \sqrt{||r'(x_{i2}) - p(x_{i1})||_2} + \\ \beta * \sum_i L_{cls}\left(f_o\left(r'^{(x_{i1} = r'(x_{i2}))},\ \ r'(x_{i2})\right), a_i\right) + \\ \gamma * \sum_i L_{cls}\left(f_r\left(r'^{(x_{i1} = r'(x_{i2}))}\right), u_i\right) \text{where } \alpha + \beta + \gamma \end{aligned} \tag{4}$$

Step 5: The refinement of the first orientation classification subnetwork as well as the object classification subnetwork of the reference network is then followed by other subnetworks being kept constant. The refinement is based on all the sample data available from the second orientation sample data labelled with their first orientation classes and their object classes. For more details, refer to Step 3.

Step 6: The training of the third orientation network and the refinement of the first and second orientation classes and object classes are conducted in the same way as in Steps 4 and 5.

3. Experimental Verification

In this study, experiments were conducted to evaluate the performance of the proposed 3D POCO Net for 3D object classification and orientation estimation. For training and testing 3D POCO Net, the ModelNet10 dataset of 3D objects was used as the reference samples. First, 4905 3D object samples were obtained from the ModelNet10 dataset as reference samples, out of which 3994 and 911 samples were selected, respectively, for training and testing. Then, the reference samples from the ModelNet10 dataset were used to generate a large pool of data for three-axis orientation variations. Specifically, we generated a 3D object dataset for x-axis, y-axis and z-axis orientation variations by rotating the reference samples about x-axis, y-axis and z-axis by (α°, 0°, 0°), (α°, α°, 0°) and (α°, α°, α°), with α chosen as 3°, 5° and 10° resolutions to cover 90° rotations. This led to a total of 152,055, 93,195 and 49,050 samples for (3°, 0°, 0°), (5°, 0°, 0°) and (10°, 0°, 0°), respectively; 4,713,705, 1,770,705 and 490,500 samples for (3°, 3°, 0°), (5°, 5°, 0°) and (10°, 10°, 0°), respectively; and 146,124,855, 33,643,395 and 4,905,000 samples for (3°, 3°, 3°), (5°, 5°, 5°) and (10°, 10°, 10°), respectively. Total training time was about 16 h for training (3°, 3°, 3°), (5°, 5°, 5°) and (10°, 10°, 10°) resolutions simultaneously up to 3200 epochs with 4 GPUs (RTX 2080 Ti). The average running time for (3°, 3°, 3°), (5°, 5°, 5°) and (10°, 10°, 10°) resolution was about 0.01 sec. for each.

Here, orientation classes are defined based on the resolution of orientation variations. However, the orientation estimation is done by the weighted sum of the orientation classes with the weights from the probability distribution of individual orientation classes so as to obtain continuous orientation estimation by regression. In addition, as a means of quantifying the performance of orientation estimation, we defined the following two performance indices, (1) the mean absolute error applied to the total testing samples, MAE-T, and (2) the mean absolute error applied only to the misclassified testing samples, MAE-F, as follows:

$$\text{MAE-T} = \frac{1}{N}\sum_i^N \left| orientation\_angle_i - u_i \right| \tag{5}$$

$$\text{MAE-F} = \frac{1}{M}\sum_j^M \left| orientation\_angle_j - u_j \right| \tag{6}$$

where u_i, u_j, N and M represent the ground truth orientation angles and the number of total and misclassified samples, respectively.

For implementation, TensorFlow on TITAN X Pascal GPU is used. In training, the Adam optimizer is used with the mini-batch size of 32, at a learning rate of 0.001.

3.1. Training and Testing of Reference Subnetwork for Object Classification Based on Samples with Reference Orientations

The reference network is trained and tested for object classification, using 3994 samples for training and 911 samples for testing. These samples serve as reference samples with reference orientations such that they are subjected to progressive rotation about z-axis, y-axis and x-axis in order to generate a larger pool of samples representing orientation variations based on the progressive framework. Note that it is free to choose the order of progressive rotations. We achieved 97.6% accuracy in object classification for the reference subnetwork with the reference samples (Table 2).

Table 2. Object classification accuracy for the reference orientation with ModelNet10 data as reference samples.

	Reference Orientation
3D POCO Net	97.6%

3.2. Training and Testing of the 1st Orientation Subnetwork for z-Axis orientation Variations

After training of the reference network is done, the first orientation network is trained and tested by defining 31, 19 and 10 orientation classes, respectively, with 3°, 5° and 10° resolutions. To this end, we augmented the reference samples by rotating them about the z-axis by 3°, 5° and 10° resolutions, as illustrated in Figure 4 in the case of 10° resolution.

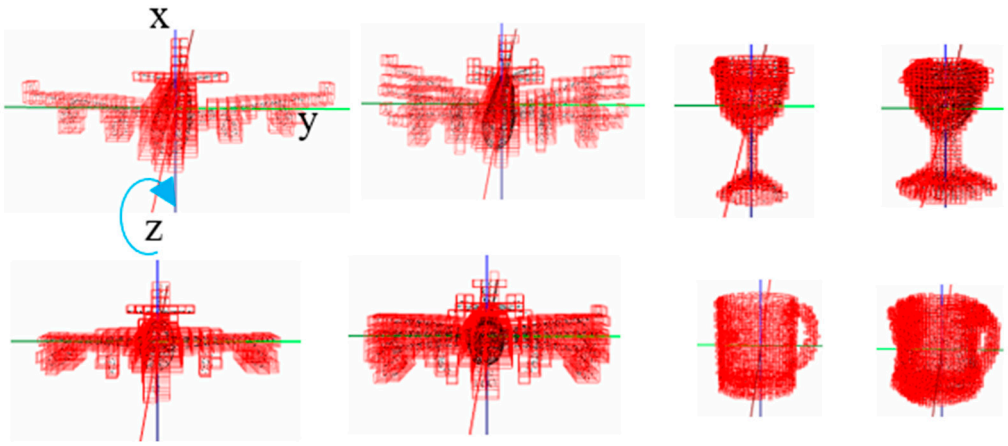

Figure 4. Examples of z-axis orientation variations (even column) by rotating reference orientation variations (odd column) around z-axis by $-10°, 0°, 10°$.

This results in 123,814, 75,886 and 39,940 samples for training and 28,241, 17,309 and 9110 samples for testing, respectively, for 3°, 5° and 10° resolutions. Tables 3 and 4 summarize the results. The result show that we can achieve less than 0.5° of MAE-T and 2.9° of MAE-F in regression with 3° precision in orientation classification, while achieving over 93% accuracy in object classification after retraining.

Table 3. Orientation classification and regression accuracy for z-axis orientation variations from ModelNet10.

	TEST3 (3°)	TEST3 (5°)	TEST3 (10°)
Orientation Classification Rate	91.6%	93.9%	98.3%
Orientation Regression MAE-T	0.44°	0.6°	0.44°
Orientation Regression MAE-F	2.89°	5.0°	12°

Table 4. Object classification accuracy for z-axis orientation variations from ModelNet10.

	TEST3 (3°)	TEST3 (5°)	TEST3 (10°)
3D POCO Net	93.7%	93.8%	93.9%

3.3. Training and Testing of the 2nd Orientation Subnetwork for z-Axis and y-Axis Orientation Variations

To train and test z-axis and y-axis orientation variations with the second orientation network, we rotate the 3D point cloud samples used for the first orientation network around the y-axis by 3°, 5° and 10° resolutions and define 961, 361 and 100 orientation classes, respectively. Figure 5 illustrates the 1st and 2nd rotations. This results in 3,838,234, 1,441,834 and 399,400 samples for training and 875,471, 328,871 and 91,100 samples for testing, respectively, for 3°, 5° and 10° resolutions.

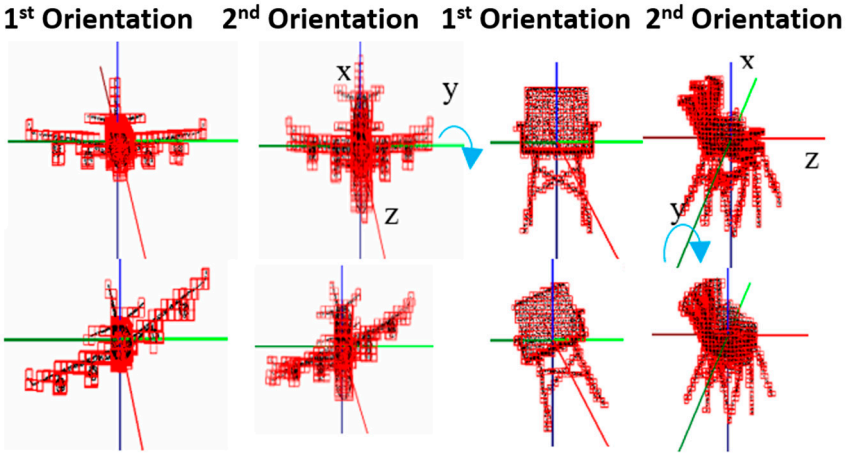

Figure 5. Examples of z-axis and y-axis orientation variations (even column) by rotating z-axis orientation variations (odd column) around y-axis by −20° and 20°.

Tables 5 and 6 show the accuracies of orientation classification and of regression with MAE-T and MAE-F for the second orientation network, as well as those of the retrained first orientation network. Table 7 shows the accuracy of object classification of the reference network after retraining. The results show that we can achieve less than 0.4° of MAE-T, 3.4 of MAE-F in orientation regression with 3° precision in orientation classification, while achieving about 93% accuracy for object classification after retraining. Although a slight degradation in performance is observed for the z-axis and y-axis orientation variations compared to only the z-axis orientation variation, the results summarized in Tables 5–7 suggest that the proposed progressive framework of learning orientation classes through data expansion and retraining works well.

Table 5. Orientation classification and regression accuracy for y-axis with z- and y-axis orientation variations for the second orientation network.

	TEST3 (3°)	**TEST3 (5°)**	**TEST3 (10°)**
Orientation Classification Rate	95.0%	96.8%	98.4%
Orientation Regression MAE-T	0.22°	0.23°	0.27°
Orientation Regression MAE-F	3.17°	5.4°	13.3°

Table 6. Orientation classification and regression accuracy for z-axis with z- and y-axis orientation variations for the first orientation network after retraining.

	TEST3 (3°)	**TEST3 (5°)**	**TEST3 (10°)**
Orientation Classification Rate	90.8%	93.8%	96.5%
Orientation Regression MAE-T	0.40°	0.47°	0.6°
Orientation Regression MAE-F	3.4°	5.9°	13.7°

Table 7. Object classification accuracy with z- and y-axis orientation variations from ModelNet10.

	TEST3 (3°)	**TEST3 (5°)**	**TEST3 (10°)**
3D POCO Net	93.0%	92.3%	92.1%

3.4. Training and Testing of the 3rd Orientation Subnetwork for z-Axis, y-Axis and x-Axis Orientation Variations

Using the z-axis and y-axis orientation variations as the reference samples, we rotate them again around x-axis with 3°, 5° and 10° resolutions to complete three-axis orientation variations for training and testing. Figure 6 illustrates 2nd and 3rd rotation examples.

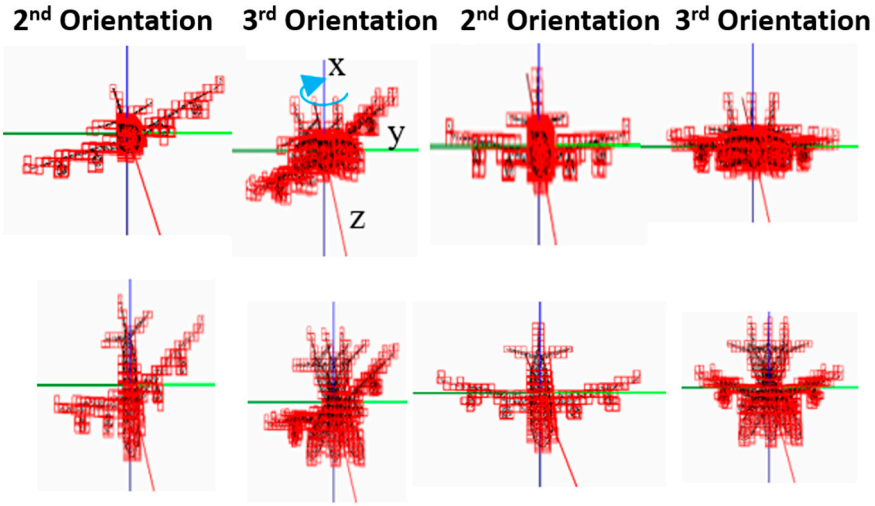

Figure 6. Examples of z-axis, y-axis and x-axis orientation variations (even column) by rotating z-axis and y-axis orientation variations (odd column) around x-axis by $-20°, 0°, 20°$.

This defines 29,791, 6859 and 1000 classes, respectively, with 3°, 5° and 10° resolutions, while augmenting training and testing samples to 118,985,254, 27,394,846 and 3,994,000 samples for training and 27,139,601, 6,248,549 and 911,000 samples for testing, respectively, for 3°, 5° and 10° resolutions. Tables 8–11 summarize testing accuracies in estimating general three-axis orientations and object class based on classification and regression for 3°, 5° and 10° resolutions. Tables 8–10 show that we can achieve the accuracies in x-, y- and z-axis orientation estimation, respectively, with 4.1°, 3.3° and 2.5° of MAE-T regression errors based on 3° resolution in orientation classification. Notice that

the accuracies in x-, y- and z-axis orientation estimation based on 5° and 10° resolutions in orientation classification are not much different from those based on 3° resolution in orientation classification. On the other hand, Table 11 shows that we can achieve about 90% in object classification accuracy, similarly for 3°, 5° and 10° resolutions in orientation classification. Note that the performance of the three orientation networks after inclusion of the 3rd orientation network for x-axis rotation is somewhat reduced, compared to that of the two orientation networks before inclusion of the 3rd orientation network. This is partly due to the fact that the ModelNet10 dataset includes some objects that are rotation-symmetric about x-axis, as illustrated in Figure 7, such that the 3rd orientation network is unable to uniquely represent x-axis orientations for those objects.

Table 8. Orientation classification and regression accuracy for x-axis with z-, y- and x-axis orientation variations for the third orientation network.

	TEST3 (3°)	**TEST**3 (5°)	**TEST**3 (10°)
Orientation Classification Rate	79.2%	85.2%	88.6%
Orientation Regression MAE-T	4.1°	4.2°	4.5°
Orientation Regression MAE-F	19.3°	23.8°	28.1°

Table 9. Orientation classification and regression accuracy for y-axis with z-, y- and x-axes orientation variations for the second orientation network after retraining.

	TEST3 (3°)	**TEST**3 (5°)	**TEST**3 (10°)
Orientation Classification Rate	79.2%	80.9%	82.3%
Orientation Regression MAE-T	3.3°	3.4°	3.6°
Orientation Regression MAE-F	6.9°	8.0°	16.2°

Table 10. Orientation classification and regression accuracy for z-axis with z-, y- and x-axis orientation variations for the first orientation network after retraining.

	TEST3 (3°)	**TEST**3 (5°)	**TEST**3 (10°)
Orientation Classification Rate	75.8%	79.1%	81.3%
Orientation Regression MAE-T	2.5°	2.8°	3.1°
Orientation Regression MAE-F	3.2°	5.8°	13.5°

Table 11. Object classification accuracy with z-, y- and x-axis orientation variations from ModelNet10.

	TEST3 (3°)	**TEST**3 (5°)	**TEST**3 (10°)
3D POCO Net	89.9%	90.2%	90.8%

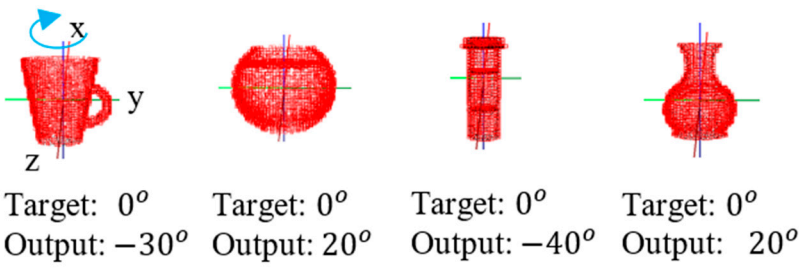

Figure 7. Examples of x-axis symmetric objects in ModelNet10 that cause incorrect orientation classification.

3.5. 3D POCO Net as a Representation Platform Applied to a Partial View 3D Point Cloud Data

The pre-trained 3D POCO Net can be used as an orientation representation platform to which additional networks are attached to solve novel 3D pose estimation problems.

To show this, we generated a partial view 3D point cloud dataset from the ModelNet10 dataset for use in partial point cloud–based object classification and orientation estimation (Figure 8). For classification of object class and orientations based on partial view 3D point clouds, we attached a PointNet, "Partial View Orientation Network," to the third orientation network of 3D POCO Net through an association subnetwork, as shown in Figure 9 (in red marks).

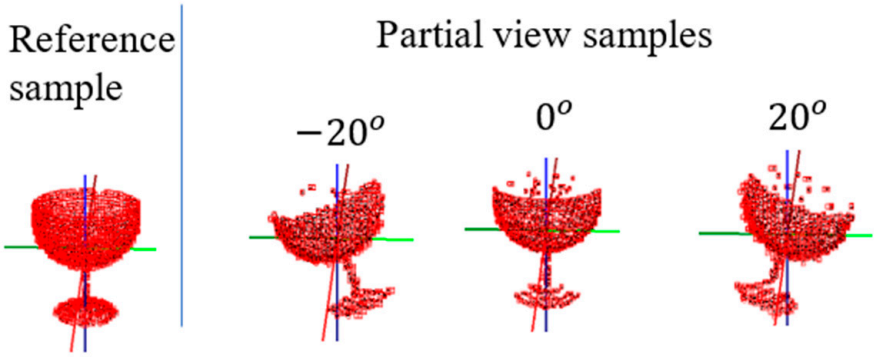

Figure 8. Partial view 3D point clouds created from full point cloud.

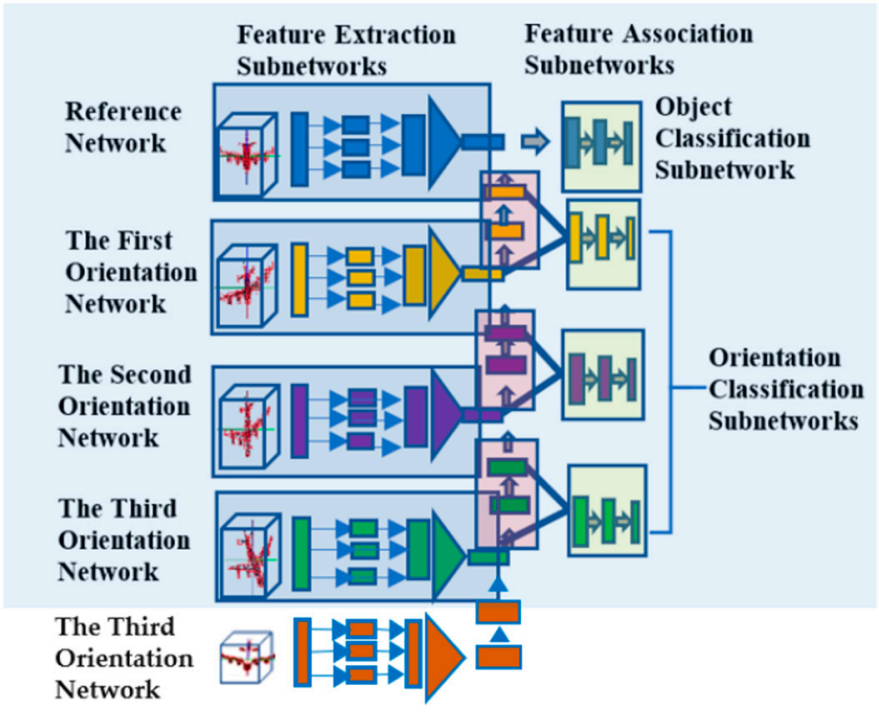

Figure 9. Expansion of 3D POCO Net to partial view data.

The partial view orientation network and the association subnetwork attached to it are trained with the pre-trained 3D POCO Net, so that the two global features, i.e., the partial view orientation network and the third orientation network of 3D POCO Net match. Notably, in training, the loss function includes the errors from all the independent networks of 3D POCO Net.

Table 12 presents the results of testing partial view data with 10 orientation classes using 10° resolution around the z-axis. As shown, we achieved 1.73°, 0.25° and 0.3° of MAE-T in regression error for the respective z-, y- and x-axis orientation estimations, while achieving about 82% accuracy for object classification. This application demonstrates the modular extensibility of the proposed 3D POCO Net as a representation platform.

Table 12. Orientation classification and regression accuracy of z-, y- and x-axis orientation variations in all three-axis classification from ModelNet10 with partial view samples in 10° resolution.

	First Orientation Network (z-Axis)	Second Orientation Network (y-Axis)	Third Orientation Network (x-Axis)
Orientation Classification Rate	92.5%	99.3%	99.3%
Orientation Regression MAE-T	1.73°	0.25°	0.3°
Orientation Regression MAE-F	13.2°	7.3°	23.9°

3.6. Discussion

The proposed progressive framework is trained to learn z-axis, y-axis and x-axis orientation variations progressively based on in-between feature associations with training samples limited only to pertinent orientation variations. Refer to Section 1.2 for the implication of the proposed progressive framework of learning object orientations in comparison with conventional deep learning approaches with fanned, grouped and hierarchical structures. We compared the performance of the proposed 3D POCO Net with that of the state-of-the-art approaches to orientation estimation (Table 13).

Table 13. Comparison of the proposed 3D POCO Net with the state-of-the-art approaches to object orientation estimation.

Reference	Approach	Dataset	No. of Classes/ Orientation Resolution	Classification Accuracy	Regression Error
Self-Supervised Learning of Point Clouds via Orientation Estimation [19]	Classification(Point Cloud-based /Fanned)	ModelNet-40	18 Spatial Vectors 32 Spatial Vectors	89.0% 90.3%	NA
Orientation-boosted Voxel Nets for 3D Object Recognition [10]	Classification (Voxel-based /Fanned)	ModelNet-10	18/20° (z-axis rot.)	89%	NA
3D Pose Regression using CNN [9]	Regression (Image-based/ Hierarchical)	Pascal 3D+	NA	NA	15.38°
Deep Learning for Spacecraft Pose Estimation [37]	Classification Based Regression (Image-based /Fanned)	Unreal Rendered Spacecraft On-Orbit (URSO)	32 × 32 × 32/30° (Euler Angles)	NA	7.4°
Our Approach: 3D POCO Net	Classification with Regression (Point Cloud-based/ Progressive)	ModelNet-10	31 × 31 × 31/3° 19 × 19 × 19/5° 11 × 11 × 11/10°	79.2% 85.2% 88.6%	2.5° 2.8° 3.1°

Due to differences in the training and testing datasets, as well as in the performance metrics, used by different approaches, direct comparison of performance is not feasible. However, Table 13 is intended to provide a general idea of where, among the current state-of-the-art approaches, the proposed 3D POCO Net with a progressive structure is positioned in terms of its methodology and performance. To further facilitate the performance assessment for the proposed approach, we extended the experiment with the ModelNet40 dataset to assess 3D POCO Net in terms of its effectiveness in handling a larger dataset. Out of total 12,308 samples in 40 object categories, we assigned 9840 and 2468 samples, respectively, to training and testing. To reduce computational burden, we defined output classes only with 10° orientation resolution for experiment. This leads to 10, 100 and 1000 output classes, respectively, defined for (10°, 0°, 0°): z-axis orientation variation, (10°, 10°,

0°): z- and y-axis orientation variations and (10°, 10°, 10°): z-, y- and x-axis orientation variations. Then, the ModelNet40 reference samples are augmented to 98,400 training and 24,680 testing samples for (10°, 0°, 0°), 984,000 training and 246,800 testing samples for (10°, 10°, 0°) and 9,840,000 training and 2,468,000 testing samples for (10°, 10°, 10°). The total training time is about 11 h for 2300 epochs, and the average running time for testing is 0.02 sec. The testing results are summarized in Table 14.

Table 14. Orientation regression and object classification accuracy of 3D POCO Net with the progressive variations of z-, y- and x-axis orientations and retraining, where 10° orientation resolution is applied to the ModelNet40 dataset.

10° Orientation Resolution	Reference Net	z-Axis Orient. Net	z- and y-Axes Orient. Net		z-, y- and x-Axes Orient. Net		
		z-Axis	z-Axis	y-Axis	z-Axis	y-Axis	x-Axis
Oient. Class. Rate	n/a	85.7%	83.2%	85.8%	75.9%	78.0%	82.7%
Ori. Regress. MAE-T	n/a	0.89°	1.2°	0.81°	5.4°	5.9°	7.2°
Ori. Regress. MAE-F	n/a	16°	15.1°	14.9°	16.2°	21.8°	30.9°
Object Class. Rate	87.3%	81.6%	80.2%		78.3%		

Table 14 shows that 3D POCO Net performs equally well with a larger dataset. The proposed framework offers a new approach for representing and estimating orientation variations, while enhancing the accuracy in orientation estimation with proper learning of in-between feature associations but with better efficiency.

4. Conclusions

In this paper, we propose a progressive deep learning framework for representing 3D objects in terms of their classes and orientations. The aim of the proposed framework is to offer high accuracy in regression and classification for three-axis orientations, yet with efficiency in the network structure. The unique features associated with the proposed framework include the following: (1) the proposed in-between association subnetworks learn to link between the networks that represent independent axes of variables so as to progressively reduce the constraint one after another; (2) independent networks are subject to retraining for refinement as the amount of data are increased with the progress of constraint relaxation. The experimental results based on the ModelNet10 dataset indicate that the proposed 3D POCO Net is effective for representing and estimating three axes of orientations with high accuracy yet with structural efficiency. For instance, the proposed 3D POCO Net is able to achieve a regression error of less than 3° in MAE-T, while achieving about 90% accuracy in object classification, for general three-axis orientation estimation and object classification with only 72 orientation output classes. The effectiveness of 3D POCO Net is further verified by applying it to general three-axis orientation estimation for a larger dataset, the ModelNet40 dataset, and for partial view-point cloud data. In particular, the latter indicates that a pre-trained 3D POCO Net can serve as an orientation representation platform to which partial point clouds from occluded 3D objects are linked for object classification and orientation estimation in a form of transfer learning. In future, further investigations will be conducted to enhance the performance by extending the applications to dealing with rotation-symmetric objects and occluded objects with a larger scale of various 3D object datasets.

Author Contributions: S.L. contributed to conceptualization, methodology development and preparation of writing. Y.Y. contributed to experimentation. Both authors have read and agreed to the published version of the manuscript.

Funding: This research was funded, in part, by "Deep Learning Based Cross-Sensory Transfer for Visually Impaired" Project (2020R1A2C200956811), of National Research Foundation (NRF), in part, by AI Graduate School Program, Grant No. 2019-0-00421, with the sponsorship by the Korean Ministry of Science and Information Technology (MSIT), and, in part, by "Smart Patient Isolation

Transport Unit System" Project (HW20C2077020020) of NRF with the sponsorship by Korea Health Industry Development Institute(KHIDI).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to acknowledge the initial contributions by Tuan Ahn Nguyen and Wencan Cheng for the work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
2. Yang, Y.; Feng, C.; Shen, Y.; Tian, D. FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018), Salt Lake City, UT, USA, 18–22 June 2018.
3. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5105–5114.
4. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
5. Xiang, Y.; Mottaghi, R.; Savarese, S. Beyond PASCAL: A benchmark for 3D object detection in the wild. In Proceedings of the Workshop on Applications of Computer Vision, Steamboat Springs, CO, USA, 24–26 March 2014; pp. 75–82.
6. Yi, L.; Kim, V.G.; Ceylan, D.; Shen, I.; Yan, M.; Su, H.; Lu, C.; Huang, Q.; Sheffer, A.; Guibas, L.; et al. A scalable active framework for region annotation in 3d shape collections. *ACM Trans. Graph. (TOG)* **2016**, *35*, 210. [CrossRef]
7. Hinterstoisser, S.; Holzer, S.; Cagniart, C.; Ilic, S.; Konolige, K.; Navab, N.; Lepetit, V. Multimodal Templates for Real-Time Detection of Texture-Less Objects in Heavily Cluttered Scenes. In Proceedings of the 2011 International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 858–865.
8. Suchi, M.; Patten, T.; Fischinger, D.; Vincze, M. Easylabel: A Semi-Automatic Pixel-wise Object Annotation Tool for Creating Robotic RGB-D Datasets. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 6678–6684.
9. Mahendran, S.; Ali, H.; Vidal, R. 3D pose regression using convolutional neural networks. In Proceedings of the International Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 494–495.
10. Sedaghat, N.; Zolfaghari, M.; Amiri, E.; Brox, T. Orientation-boosted voxel nets for 3D object recognition. In Proceedings of the 28th British Machine Vision Conference, London, UK, 4–7 September 2017.
11. Qi, C.R.; Su, H.; Nießner, M.; Dai, A.; Yan, M.; Guibas, L.J. Volumetric and multi-view cnns for object classification on 3d data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 5648–5656.
12. Kiru, P.; Timothy Patten, M.V. Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation. In Proceedings of the ICCV, Seoul, Korea, 27–28 October 2019; pp. 7668–7677.
13. Su, H.; Maji, S.; Kalogerakis, E.; Learned-Miller, E. Multi-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 13–16 December 2015; pp. 945–953.
14. Devgon, S.; Ichnowski, J.; Balakrishna, A.; Zhang, H.; Goldberg, K. Orienting Novel 3D Objects Using Self-Supervised Learning of Rotation Transforms. In Proceedings of the 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), Hong Kong, China, 20–21 August 2020; pp. 1453–1460.
15. Wu, J.; Zhang, C.; Xue, T.; Freeman, B.; Tenenbaum, J. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 82–90.
16. Maturana, D.; Scherer, S. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015.
17. Brock, A.; Lim, T.; Ritchie, J.M.; Weston, N. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv* **2016**, arXiv:1608.04236.
18. Riegler, G.; Osman Ulusoy, A.; Geiger, A. Octnet: Learning deep 3d representations at high resolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3577–3586.
19. Poursaeed, O.; Jiang, T.; Qiao, H.; Xu, N.; Kim, V.G. Self-Supervised Learning of Point Clouds via Orientation Estimation. In Proceedings of the 2020 International Conference on 3D Vision (3DV), Fukuoka, Japan, 25–28 November 2020; pp. 1018–1028.

20. Weibel, J.B.; Patten, T.; Vincze, M. Robust 3D object classification by combining point pair features and graph convolution. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 7262–7268.
21. Hanocka, R.; Hertz, A.; Fish, N.; Giryes, R.; Fleishman, S.; Cohen-Or, D. MeshCNN: A network with an edge. *ACM Trans. Graph.* **2019**, *38*, 1–90. [CrossRef]
22. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. In Proceedings of the International Conference on Learning Representations, ICLR, Toulon, France, 24–26 April 2017.
23. Hinterstoisser, S.; Lepetit, V.; Ilic, S.; Holzer, S.; Bradski, G.; Konolige, K.; Navab, N. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. Available online: http://www.stefan-hinterstoisser.com/papers/hinterstoisser2012accv.pdf (accessed on 2 May 2020).
24. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [CrossRef] [PubMed]
25. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
26. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Berlin, Germany, 2016; pp. 21–37.
27. Timmermann, D.; Heid, D.; Friedel, T.; Roennau, A.; Dillmann, R. A Hybrid Approach for Object Localization Combining Mask R-CNN and Halcon in an Assembly Scenario. In Proceedings of the 2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA), Chengdu, China, 23-26 April 2021; pp. 270–276.
28. Xiang, Y.; Schmidt, T.; Narayanan, V.; Fox, D. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In Proceedings of the Robotics: Science and System XIV, Pittsburgh, PA, USA, 26–30 June 2018.
29. Wang, C.; Xu, D.; Zhu, Y.; Martín-Martín, R.; Lu, C.; Fei-Fei, L.; Savarese, S. DenseFusion: 6D object pose estimation by iterative dense fusion. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–21 June 2019. in press.
30. Chen, X.; Dong, Z.; Song, J.; Geiger, A.; Hilliges, O. Category Level Object Pose Estimation via Neural Analysis-by-Synthesis. In Proceedings of the European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020; Volume 26, pp. 139–156.
31. Wang, H.; Sridhar, S.; Huang, J.; Valentin, J.; Song, S.; Guibas, L.J. Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2019, Long Beach, CA, USA, 15–21 June 2019; pp. 2642–2651.
32. Rusu, A.A.; Rabinowitz, N.C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; Hadsell, R. Progressive neural networks. *arXiv* **2016**, arXiv:1606.04671.
33. Zhang, Z.; Ning, G.; Cen, Y.; Li, Y.; Zhao, Z.; Sun, H.; He, Z. Progressive Neural Networks for Image Classification. *arXiv* **2018**, arXiv:1804.09803.
34. Moriya, T.; Masumura, R.; Asami, T.; Shinohara, Y.; Delcroix, M.; Yamaguchi, Y.; Aono, Y. Progressive Neural Network-based Knowledge Transfer in Acoustic Models. In Proceedings of the 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Honolulu, HI, USA, 12–15 November 2018; pp. 998–1002.
35. Gideon, J.; Khorram, S.; Aldeneh, Z.; Dimitriadis, D.; Provost, E.M. Progressive Neural Networks for Transfer Learning in Emotion Recognition. *arXiv* **2017**, arXiv:1706.03256.
36. Fu, R.; Tao, J.; Zheng, Y.; Wen, Z. Transfer Learning Based Progressive Neural Networks for Acoustic Modeling in Statistical Parametric Speech Synthesis. In Proceedings of the ISCA Interspeech, Hyderabad, India, 2–6 September 2018; pp. 907–911.
37. Proenca, P.F.; Gao, Y. Deep Learning for Spacecraft Pose Estimation from Photorealistic Rendering. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 6007–6013.
38. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.