DepthVoting: A Few-Shot Point Cloud Classification Model Incorporating a Projection-Based Voting Mechanism

Yunhui Zhu, Jiajing Chen, Senem Velipasalar
Syracuse University, Electrical Engineering and Computer Science Dept.,
Syracuse, NY, USA
{yzhu130, jchen152, svelipas}@syr.edu *

Abstract

Despite the significant progress in few-shot 2D image classification, few-shot 3D point cloud classification remains relatively under-explored, particularly in addressing the challenges posed by missing points in 3D point clouds. Most existing methods for few-shot 3D point cloud classification are point-based, and thus, highly sensitive to missing points. Despite recent attempts, such as ViewNet, which introduce projection-based backbones to increase robustness against missing points, the reliance on max pooling, to extract information from multiple images simultaneously, makes them prone to information loss. To address these limitations, we introduce DepthVoting, a novel projection-based approach, for few-shot 3D point cloud classification. Instead of extracting features from multiple projection images simultaneously, DepthVoting captures features from pairs of projection images (obtained from opposite view angles) separately, enhancing the extraction of more comprehensive information. These features are sent to multiple few-shot heads, which share parameters. To further refine predictions, DepthVoting incorporates a voting mechanism, allowing contribution and incorporating information from different pairs. We conduct extensive experiments on three datasets, namely ModelNet40, ModelNet40-C, and ScanObjectNN, along with cross-validation. Our proposed method consistently outperforms the state-of-the-art baselines on all datasets in terms of average accuracy with even higher margins on the challenging ScanObjectNN dataset.

1. Introduction

In recent years, deep learning-based 3D point cloud analysis has attracted increasing attention from academia and industry, due to the use of 3D point clouds in wide range of applications, such as autonomous driving, robotics, and simultaneous localization and mapping (SLAM), and more availability of data thanks to increasing accessibility of sensors, such as LiDAR. Although significant amount of work has been performed on 2D few-shot learning, and fully supervised point cloud classification, few-shot learning (FSL) from point clouds still remains under-explored.

Different from 2D images, 3D point cloud data is unstructured, making convolutional neural networks (CNN) inapplicable. To address this issue, many point-based methods [9, 10, 20] were proposed for fully supervised point cloud analysis. As for few-shot point cloud classification, Ye *et al*. [22] adopt DGCNN [20], a point-based method, as the backbone. However, Chen *et al*. [2] show that point-based backbones are sensitive to missing points in point clouds, and thus, are not the most suitable for FSL tasks. They discuss that some projection depth images generated from a point cloud are more robust missing points, and propose ViewNet [2], which employs features extracted from six projections images for FSL. They report better performance than point-based backbones.

Although ViewNet [2] reported state-of-the-art (SOTA) performance on several datasets, we argue that it cannot fully leverage the information contained in depth images. ViewNet employs max-pooling among features of all six projections, which may lead to information loss especially when the number of projection images increases. To motivate our claim, we perform an experiment in Sec. 3, showing that, in some folds, a single pair of projection images processed through ResNet18 can outperform ViewNet, which uses a total of six projections.

To address the aforementioned challenges and motivated by our findings, we introduce a novel projection-based method, referred to as the DepthVoting, for 3D point cloud classification. DepthVoting employs a projection method adopted from SimpleView [5] to generate six 2D depth images, corresponding to front, back, top, bottom, left, and right views, from 3D point clouds. To enrich the amount of information, data augmentation is performed by rotating all projection images 180-degree clockwise, yielding 12 im-

*This work was supported in part by New York State Department of Economic Development under Contract Number C080116.

ages as input. DepthVoting adopts ResNet-18 [17] as the backbone for feature extraction. In contrast to using conventional average pooling or max pooling, which are prone to information loss, DepthVoting employs set pooling to preserve a more comprehensive set of features. Recognizing the similarities among the opposite views (top and bottom, left and right, and front and back), proposed DepthVoting model partitions feature maps into three sub-feature maps corresponding to each pair. Notably, as discussed in Sec. 3, only single pairs of projection images can provide relatively good performance by themselves, even surpassing ViewNet for some folds in few-shot point cloud classification. Combining this observation with the motivation to avoid max pooling, we employ three few-shot heads (sharing the same parameters) to process each sub-feature map separately, enabling independent learning from each sub-feature map. Subsequently, DepthVoting integrates a voting mechanism, to obtain the final class prediction for a query.

Experiments conducted on the ModelNet40 [21], ModelNet40-C [15] and ScanObjectNN [18] datasets, utilizing cross-validation, highlight the success of our method compared to five different SOTA baselines in the few-shot 3D point cloud classification task. The main contributions of this work include the following:

- We provide motivation showing that a single pair of depth images projected from 3D point cloud data can provide good performance by themselves, even surpassing ViewNet for some folds.
- Motivated by our findings, we propose DepthVoting, which extracts features from individual projection pairs, and sends them to multiple parameter-sharing few-shot heads to separately learn their features, instead of employing view pooling. DepthVoting also incorporates a soft voting mechanism, providing improved accuracy for few-shot point cloud classification.
- DepthVoting achieves SOTA performance by surpassing five different baselines [2, 7, 13, 16, 22] on all of three datasets. The improvements in accuracy, and higher improvement margins on the challenging ScanObjectNN dataset serve as the strong evidence of our model's effectiveness when dealing with real-world point cloud scans.
- We present a series of ablation studies to further show the effectiveness of DepthVoting, and the contribution and importance of using multiple parameter-sharing few-shot heads and a voting mechanism.

2. Related Work

2.1. Point Cloud Classification

A 3D point cloud is a set of unordered points, represented by their x, y and z coordinates. Some previous works [9, 10] perform point cloud classification by extracting features directly from 3D points. In contrast, other existing works [3, 14] first project a point cloud onto depth images so that existing CNNs can be adopted to extract image features. In general, point cloud classification methods can be broadly categorized as the projection-based and point-based techniques.

Projection-based methods. Projection-based methods focus on transforming unstructured 3D point cloud data into structured formats for feature extraction. MVCNN [14] presents a multi-view approach that uses 12 2D images obtained from projecting a 3D point cloud. Subsequently, a CNN is employed to extract features from these images. These features are then combined through view pooling, which aggregates the information captured by different views. Finally, an additional CNN is employed to perform the ultimate classification task. Since different views can have different contributions to the classification task, GVCNN [3] assigns distinct scores to individual view descriptors and categorizes these descriptors into separate groups based on their scores. Subsequently, the method generates weights for each group, which are used for weighted aggregation to obtain the final features.

Point-based methods. Different from the projection-based approaches, point-based methods focus on learning features directly from 3D points. PointNet [9] employs max pooling to obtain permutation invariant features. However, PointNet extracts local features only from individual points, without incorporating global information. To address this issue, various methods were proposed to extract both local features, from individual points, and global features, based on the relationships among points. PointNet++ [10] is a pioneering method in this direction, which uses a hierarchical neural network structure. It progressively partitions a point cloud into local regions, learning features at multiple scales. Another notable approach is DGCNN [20], which builds a graph, where each point is connected to its k-nearest neighbors. By using these local graphs, DGCNN captures both local and global structures of the point cloud data.

2.2. Few-shot Learning

Since the process of collecting, labeling, and validating large datasets can be prohibitively expensive, few-shot learning (FSL) was proposed to address this problem. This learning paradigm aims to train a model that can generalize to new classes, which are not seen during training, and perform prediction given only a few labeled samples.

In FSL, the dataset is divided as $D = (D^{base}, D^{novel})$, where D^{base} includes all the examples used for training and D^{novel} contains the examples used in the testing stage. The classes in D^{base} do not overlap with the classes in D^{novel}. The framework involves a support set and a query set. The support set is built by randomly selecting N classes from the dataset and choosing K samples for each selected class, resulting in a total of $N \times K$ samples. The query set is generated by randomly selecting Q samples from the re-

maining data of the same N classes used for the support set, resulting in another subset of $N \times Q$ samples. The goal of the model is to classify the query samples given $N \times K$ support samples. This task is referred to as the N-way K-shot Q-query task. FSL often adopts meta-learning to quickly adapt to new tasks. Few-shot meta-learning can generally be categorized into three groups: model-based, metric-based, and optimization-based methods.

Model-based Methods. Model-based methods are designed to efficiently update parameters using only a limited number of samples. MANN [12] is a pioneering technique that employs memory augmentation to address the challenges of FSL. This approach extends traditional neural networks by integrating memory and the corresponding read and write mechanisms. Similarly, MetaNet [8] utilizes an external memory to save model weights, enabling rapid parameterization for generalization. MetaNet consists of a meta learner and a base learner. The meta learner acquires generalization information across a range of meta tasks and utilizes a memory mechanism to preserve this knowledge. On the other hand, the base learner is designed to quickly adapt to new tasks and collaborates with the meta learner to generate predictions for input data.

Metric-based Methods. These methods focus on how to measure and compare similarity between data points. They compute the distance between samples in the query set and samples in the support set to perform the classification. Siamese Neural Networks [6] employ a twin network structure for classification. It uses the same network structure to extract features from two images. If the distance of two samples in the feature space is very close, those samples should belong to the same class, or vice versa. Different from Siamese Neural Networks, Match Network [19] uses different networks to extract features from query set and support set. A weighted sum of predicted values between the samples in support set and query set drives classification. Later, Prototype Network [13] was proposed, which computes class prototypes through means and subsequently classifies queries based on the distance between prototypes and query data.

Optimization-based Methods. These methods aim to enhance few-shot classification by redefining the optimization approach. Ravi *et al*. [11] present a two-level optimization process. The inner loop adapts the model's parameters for each task using stochastic gradient descent. The outer loop involves updating the initial model parameters to facilitate effective adaptation across tasks. Model-Agnostic Meta-Learning (MAML) [4] introduces a meta-learning approach that remains adaptable to any model using gradient descent. MAML exploits limited data to determine an optimal range of initial values, thereby directing gradient descent towards more responsive initial parameters. This enables rapid model fitting.

3. Motivation

Real-world point cloud data often times suffers from missing points. ViewNet [2] is a recently proposed method providing SOTA performance on few-shot point cloud classification. It employs three pairs of projection images (a total of six images), obtained from opposite view angles, to increase robustness against missing points. ViewNet forms different projection feature combinations, performs view pooling on these combinations, and reports better performance than the baselines.

Since ScanObjectNN dataset [18] is captured from real-world scans, and involves missing points, we have conducted an experiment on this dataset to assess the performance of using only a single pair of projection images (from opposite view angles) for few-shot classification. We first employ a projection method similar to SimpleView [5] to generate six 2D projection images (front, back, top, bottom, left, right). Then, different from ViewNet, we extract features, only from a single pair of images, by using ResNet-18 [17]. The classification task is executed using the Cross Instance Adaptation module (CIA) [22]. We divide the 15 classes in the ScanObjectNN dataset into three folds, with each fold containing five classes, and perform three-fold cross-validation. The results are presented in Table 1. As can be seen, single pairs of projection images can provide relatively good performance by themselves, even surpassing ViewNet for some folds. For instance, for fold 2 of the 5-way 5-shot experiment, the performance of ViewNet (77.46%), which employs all six images for prediction, is 1.52% lower than that (78.98%) of using only Top and Bottom projections with ResNet18. These results show that ViewNet cannot fully exploit the features of these individual pairs of images, and lose some information due to view pooling and max pooling of all six images. In this case, the useful information contained in the Top and Bottom views is not sufficiently leveraged by ViewNet. Motivated by these results, we present a new way of extracting and combining information from different projection image pairs, which employs a voting strategy among different pairs (instead of pooling), and enables the model to independently learn from each projection pair.

		fold 0	fold 1	fold 2	Mean
5-way 1-shot	Front + Back	48.98	61.56	62.86	57.80
	Left + Right	53.45	61.57	60.72	58.58
	Top + Bottom	53.54	63.41	**64.71**	60.55
	ViewNet [2]	**60.90**	**66.48**	64.10	**63.83**
5-way 5-shot	Front + Back	69.23	72.98	74.68	72.30
	Left + Right	67.70	71.77	72.91	70.79
	Top + Bottom	69.18	**75.27**	**78.98**	74.48
	ViewNet [2]	**73.66**	74.77	77.46	**75.30**

Table 1. Few-shot classification results using a single pairs of projection images on the ScanObjectNN dataset.

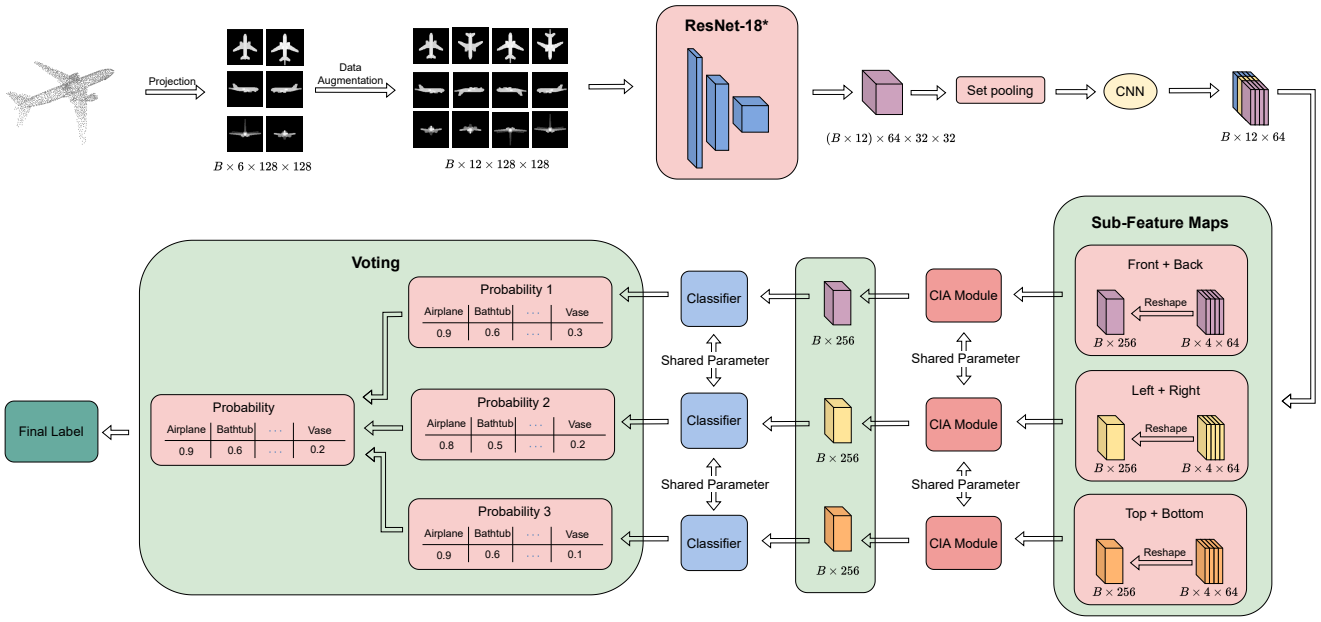

Figure 1. **Overall Architecture of DepthVoting.** Initially, the point cloud is projected onto six 2D depth images, followed by data augmentation through rotation. B represents the batch size. Then, our approach employs ResNet-18* (* signifies that our model employs part of the ResNet-18 architecture) combined with set pooling to generate a feature map for the point cloud. This feature map is divided into three sub-feature maps. To facilitate independent learning from different projection pairs, each sub-feature map is separately sent to the same CIA few-shot head. Following this, a voting mechanism is employed to consolidate the individual predictions from each sub-feature map, resulting in the final prediction for the query.

4. Proposed Method

Real-world 3D point cloud data, especially those captured outdoors with LiDAR scans, suffer from noise, missing points and occlusions. Hence, we propose DepthVoting, a novel projection-based architecture for few-shot 3D point cloud classification, which increases robustness against losing information through strategic use of information coming from different pairs of projection depth images. The architecture of DepthVoting is illustrated in Fig. 1.

We focus on N-way K-shot Q-query few-shot classification task, wherein the support set S contains K labeled samples for each of the N classes, while the query set L consists of $N \times Q$ samples, all drawn from these N classes. The objective of our model is to classify the $N \times Q$ samples into their corresponding classes.

4.1. Few-Shot Backbone

Consider a set of 3D points $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, where each point is represented as $p_i = (x_i, y_i, z_i)$. We first project the point cloud data onto six 2D depth images (corresponding to front, back, top, bottom, left, and right views) by using a projection technique similar to SimpleView [5]. This results in a tensor $P \in \mathbb{R}^{6 \times H \times W}$, where 6 is the number of 2D projection images, and H and W denote the height and width of each depth image, respectively. In order to enrich the amount of information, we employ data augmentation by rotating all 2D projection images clockwise by 180 degrees. This results in a set of 12 2D projection images, forming a tensor $P \in \mathbb{R}^{12 \times H \times W}$. An example set of 12 depth images for a point cloud from the ModelNet40 dataset is presented in Fig. 2.

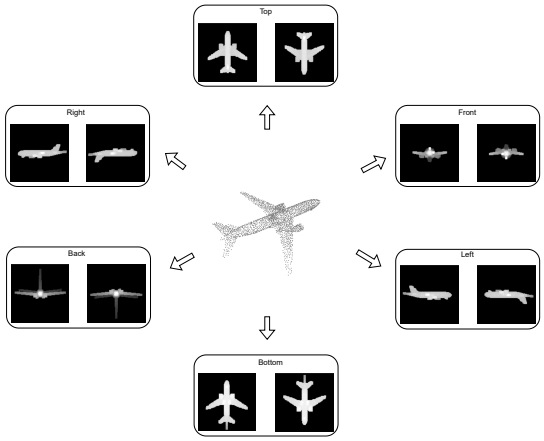

Figure 2. 2D projection images are generated by projecting a ModelNet40 point cloud onto six orthogonal planes by a projection method as in SimpleView [5]. To augment data, all six images are rotated clockwise by 180 degrees. This process results in 12 distinct 2D projection images for the point cloud.

Our DepthVoting model employs ResNet-18 [17] as the image processing backbone. This enables us to obtain feature representations from 12 projection images. In contrast to employing conventional average pooling or max-

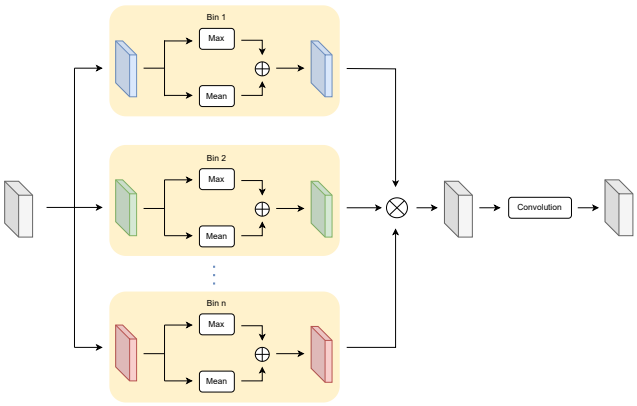

Figure 3. The structure of set pooling. '⊗' is the concatenation operation. '⊕' is the summation operation.

pooling operations at the end of ResNet-18, we adopt the set pooling mechanism. This choice can preserve a more comprehensive set of features while minimizing information loss typically associated with max pooling operations. The structure of set pooling [1, 2] is depicted in Fig. 3. Set pooling aims to capture features from diverse pixel fields. We divide the feature map $F \in \mathbb{R}^{(B\times12)\times64\times32\times32}$, obtained from ResNet-18, into b_i-many bins, where b_i is chosen from the set $b = \{1, 2, 4, 8, 16, 32\}$. Within each bin, we compute both the mean and maximum values to amalgamate local and global features for the given pixel group. Notably, the selection of the number of bins b_i, from the set $\{1, 2, 4, 8, 16, 32\}$, ensures that each bin encapsulates unique pixel fields. This choice allows us to gather a comprehensive array of information spanning various fields. Consequently, we summarize these features to yield the output of each bin denoted as $K_i \in \mathbb{R}^{(B\times12)\times64\times b_i}$, where B represents the batch size.

Ultimately, by concatenating the features from all bins, we form the consolidated feature $K \in \mathbb{R}^{(B\times12)\times64\times63}$. This feature is further processed through a series of convolutional layers to produce the final output $F \in \mathbb{R}^{B\times12\times64}$.

4.2. Few-shot Head

Our experiment, presented in Sec. 3, has shown that single pairs of 2D projection images can provide relatively good performance by themselves, even surpassing ViewNet (which uses all three pairs) for some folds. These results also reveal that ViewNet cannot fully exploit the features from individual pairs of images, and loses some information due to view pooling and max pooling of all three pairs. Motivated by this, we present a new way of extracting and combining information from different projection image pairs.

To capture distinguishing features of the point cloud and address subtle inter-class differences, we adopt the Cross Instance Adaptation module (CIA) [22] as the few-shot head, and apply it to the sub-feature maps obtained as described below.

After extracting the feature map F, using ResNet18 together with set pooling, we partition this feature map into three distinct sub-feature maps as follows: a front and back feature map denoted by $f_1 \in \mathbb{R}^{B\times4\times64}$, a top and bottom feature map denoted by $f_2 \in \mathbb{R}^{B\times4\times64}$, and a left and right feature map denoted by $f_3 \in \mathbb{R}^{B\times4\times64}$. Given the distinct nature of the three sub-feature maps, f_1, f_2, and f_3, we send them separately to the same CIA head. This approach enables the model to independently learn from each sub-feature map.

The CIA module contains two pivotal components: the Self-Channel Interaction Module and the Cross-Instance Fusion Module. The Self-Channel Interaction Module begins by generating a query-vector $q \in \mathbb{R}^{1\times d}$ and a key-vector $k \in \mathbb{R}^{1\times d}$ from the embedding feature vector f using two separate linear embedding functions. Then, a channel-wise relation score map is obtained:

$$R = q^T k, R \in \mathbb{R}^{d\times d}. \tag{1}$$

Then, the updated features are obtained by:

$$f' = f + fR', \ f' \in \mathbb{R}^{1\times d}, \tag{2}$$

where R' is computed as:

$$R'_{ij} = \frac{exp(R_{ij})}{\sum_{k=1}^{d} exp(R_{kj})}, R' \in \mathbb{R}^{d\times d}. \tag{3}$$

The Cross-Instance Fusion Module involves the concatenation of each prototype feature f_p^i with its top K_1 cosine-similar query features and the concatenation of each query feature f_q^i with its top K_2 cosine-similar prototype features. More details can be found in the original CIA paper [22].

After separately sending the sub-feature maps, f_1, f_2, and f_3 to the CIA module, we generate three distinct probabilities, namely p_1, p_2, and p_3, which quantify the likelihood of the query data belonging to each class. To determine the final class label for the query, we implement a voting mechanism, which enhances the overall robustness and reliability of the classification process.

4.3. Voting Mechanism

Voting stands out as a powerful combination strategy employed in ensemble learning to effectively address classification challenges. This technique harnesses the collective wisdom of multiple components to make a more robust and accurate prediction.

In classification tasks, there are two primary forms of voting: hard voting and soft voting. In hard voting, the ultimate prediction is made by choosing the label with the highest occurrence among all the voting outcomes. Soft voting, on the other hand, calculates the average of probabilities assigned to a specific class by multiple models and selects the prediction with the highest average probability.

In this work, we adopt the soft voting mechanism, which is well-suited for our approach. As described above, we derive three probabilities, denoted by p_1, p_2, and p_3, from each sub-feature map, for each query set's assignment to specific classes. Subsequently, we determine the final probability of the query set's assignment to particular class by averaging these probabilities as $p = \frac{p_1+p_2+p_3}{3}$.

Finally, the classes with the highest average probability are chosen as the final predicted class for a given query set. Given that we run three sub-feature maps separately through a few-shot head, we derive three individual losses from these, denoted by L_1, L_2, and L_3. To obtain the final loss L for prediction, we average these individual losses. Therefore, the final loss of the prediction is computed as:

$$L = \frac{L_1 + L_2 + L_3}{3}. \tag{4}$$

5. Experiments

We train our proposed method using the meta-learning scheme. All experiments are performed using one RTX6000 GPU. We use cross-entropy as the loss function, and employ Adam optimizer with a learning rate of 8×10^{-4} and weight decay of 0.5. We observe that 100 epochs are enough for all models to converge, including our proposed DepthVoting and five baseline methods, namely MetaOptNet [7], RelationNet [16], ProtoNet [13], CIA [22] and ViewNet [2]. Among these baselines, MetaOptNet, RelationNet, and ProtoNet have been developed for 2D few-shot learning, while CIA and ViewNet have been proposed for 3D few-shot learning. The results we report are the accuracy values calculated on test episodes with 95% confidence intervals. We train and test all models under the same settings on three commonly used datasets, namely ModelNet40, ModelNet40-C and ScanObjectNN. Experimental results are presented in Tables 2, 3, and 4, respectively, for these datasets.

5.1. Experiment Results on Modelnet40

ModelNet40 [21] is a well-established dataset, which is widely employed in few-shot tasks. It contains 12,311 CAD models from 40 categories. Each CAD model contains 1024 3D points. In Fig. 4, we present example point clouds of an airplane and a chair from the ModelNet40 dataset, along with three corresponding projected images.

For the few-shot classification task, we sort 40 classes based on their class IDs in ascending order, and divide them into 4 folds, each containing 10 classes, to perform 4-fold cross-validation. Table 2 presents the accuracy with 95% confidence intervals obtained from our proposed method and five baselines. Our method consistently outperforms all baselines, both for 1-shot and 5-shot classification. Since Modelnet40 is generated from CAD models, it is the least challenging dataset among the three, explaining the relatively high performance of most approaches, and thus, the smaller margins of improvement provided by DepthVoting.

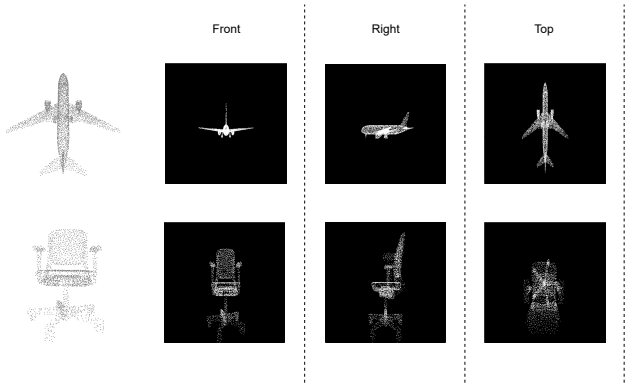

Figure 4. Example point clouds of an airplane and a chair and three of the corresponding projected images for ModelNet40 dataset.

5.2. Experiment Results on Modelnet40-C

ModelNet40-C [15] is a more recent dataset, including the same 40 classes as the ModelNet40 dataset. Unlike ModelNet40, ModelNet40-C introduces 15 corruptions, including LiDAR, occlusion, gaussian and others, each with 5 severity levels, to simulate real-world point cloud data. In Fig. 5, we present example point clouds of an airplane and a chair from the ModelNet40-C dataset, along with their corresponding three projection images. Compared to the point clouds belonging to the same class in the ModelNet40 dataset (shown in Fig. 4), point clouds in ModelNet40-C have missing points, making this dataset more challenging.

We adopt the same dataset splitting methodology used in ModelNet40. Table 3 presents the accuracy values obtained with our model and five different baselines. Our method consistently outperforms all baselines, both for 1-shot and 5-shot classification. As can be seen, all the methods experience a performance degradation on ModelNet40-C compared to ModelNet40, since ModelNet40-C is more challenging as described above.

5.3. Experiment Results on ScanObjectNN

ScanObjectNN [18] is a widely used benchmark dataset for 3D point cloud classification. It contains over 15,000 labeled point clouds, from 15 object categories, captured from real-world indoor environments. The point clouds are acquired through RGB-D cameras and LiDAR scanners. In Fig. 6, we present example point clouds of a desk and a chair from the ScanObjectNN dataset, along with their corresponding three projection images. Compared to the point clouds in ModelNet40 and Modelnet40-C datasets (shown in Fig. 4 and Fig. 5), point clouds in ScanObjectNN dataset present more challenges with a lot more missing points.

	5-way 1-shot					5-way 5-shot				
	fold 0	fold 1	fold 2	fold 3	Mean	fold 0	fold 1	fold 2	fold 3	Mean
MetaOptNet [7]	82.87±0.72	75.77±0.83	65.31±0.92	66.97±0.93	72.73±0.85	92.37±0.38	86.44±0.62	82.10±0.58	83.15±0.55	86.02±0.53
RelationNet [16]	82.14±0.69	77.46±0.80	66.09±0.91	69.47±0.84	75.23±0.81	91.53±0.38	85.11±0.61	79.36±0.63	83.01±0.52	84.75±0.53
ProtoNet [13]	85.42±0.64	79.46±0.76	70.06±0.39	70.73±0.42	76.42±0.55	93.99±0.29	88.65±0.54	84.76±0.51	85.56±0.48	88.24±0.45
CIA [22]	89.97±0.63	83.46±0.83	74.08±0.95	76.13±0.86	80.91±0.82	94.61±0.30	89.15±0.55	85.00±0.51	86.71±0.50	88.87±0.47
ViewNet [2]	**92.57±0.52**	82.68±0.80	75.28±0.90	80.95±0.75	82.87±0.74	96.23±0.26	89.64±0.55	85.74±0.51	90.18±0.45	90.45±0.44
DepthVoting (Ours)	91.99±0.57	**84.85±0.79**	**77.04±0.85**	**82.65±0.83**	**84.13±0.76**	**96.84±0.25**	**90.30±0.54**	**88.41±0.45**	**91.95±0.43**	**91.88±0.42**

Table 2. Few-shot classification results on the ModelNet40 dataset. **Bold** indicates the best result.

	5-way 1-shot					5-way 5-shot				
	fold 0	fold 1	fold 2	fold 3	Mean	fold 0	fold 1	fold 2	fold 3	Mean
MetaOptNet [7]	78.28±0.79	75.34±0.84	58.07±0.86	66.29±0.91	69.50±0.85	91.09±0.40	84.19±0.57	75.10±0.73	81.34±0.53	82.93±0.56
RelationNet [16]	79.59±0.74	74.63±0.84	59.03±0.81	68.38±0.86	70.41±0.81	87.12±0.46	83.55±0.54	70.18±0.78	79.01±0.58	79.97±0.59
ProtoNet [13]	81.29±0.71	75.83±0.79	61.76±0.84	69.83±0.84	72.18±0.80	90.97±0.39	86.21±0.50	76.99±0.65	83.19±0.51	84.34±0.51
CIA [22]	85.70±0.75	79.67±0.90	65.68±1.00	74.32±0.94	76.34±0.89	92.07±0.36	86.81±0.56	76.11±0.71	83.71±0.51	84.68±0.54
ViewNet [2]	89.47±0.58	**81.05±0.78**	69.56±0.89	**76.29±0.85**	79.09±0.78	94.95±0.31	88.75±0.49	81.53±0.60	86.78±0.46	88.00±0.47
DepthVoting (Ours)	**89.94±0.65**	79.29±0.81	**71.67±0.93**	75.54±0.91	**79.11±0.83**	**96.32±0.27**	**88.79±0.52**	**83.72±0.60**	**87.38±0.47**	**89.05±0.47**

Table 3. Few-shot classification results on the ModelNet40-C dataset. **Bold** indicates the best result.

	5-way 1-shot				5-way 5-shot			
	fold 0	fold 1	fold 2	Mean	fold 0	fold 1	fold 2	Mean
MetaOptNet [7]	41.92±0.72	61.12±0.66	53.87±0.78	52.30±0.72	63.86±0.56	67.73±0.45	70.19±0.49	67.26±0.50
RelationNet [16]	50.29±0.76	54.23±0.63	51.45±0.64	51.99±0.68	58.65±0.53	66.72±0.50	65.94±0.52	63.77±0.52
ProtoNet [13]	50.81±0.73	60.46±0.67	58.72±0.78	56.66±0.73	68.45±0.54	70.20±0.52	68.76±0.49	69.13±0.52
CIA [22]	50.58±0.82	62.17±0.68	62.59±0.74	58.45±0.75	62.94±0.51	71.31±0.45	70.21±0.48	68.15±0.48
ViewNet [2]	**60.90±0.76**	66.48±0.60	64.10±0.77	63.83±0.71	73.66±0.48	74.77±0.45	77.46±0.46	75.30±0.46
DepthVoting (Ours)	60.51±0.73	**66.80±0.71**	**69.90±0.73**	**65.74±0.72**	**77.02±0.42**	**77.21±0.45**	**84.76±0.38**	**79.66±0.42**

Table 4. Few-shot classification results on the ScanObjectNN dataset. **Bold** indicates the best result.

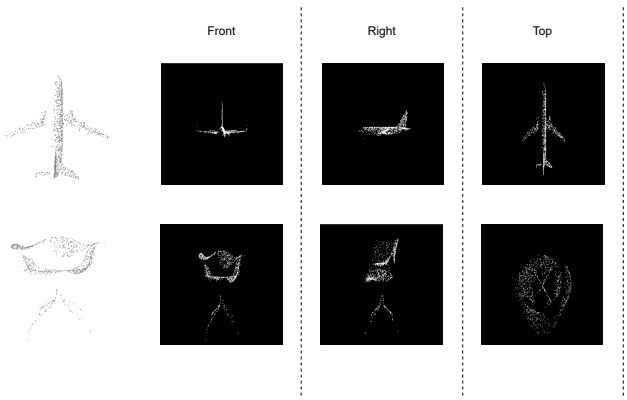

Figure 5. Example point clouds of an airplane and a chair and three of the corresponding projected images for ModelNet40-C dataset.

For the few-shot classification task, we sort the 15 classes based on their class IDs in ascending order and divide them into 3 folds, each containing 5 classes, to perform 3-fold cross-validation. Tab. 4 presents the accuracy values obtained with our model and five different baselines. Our method consistently outperforms all baselines by significant margins in all folds, both for 1-shot and 5-shot classification. More specifically, in 5-way 5-shot classification, our approach surpasses the second-best performer (ViewNet) by a margin of 4.36%. In 5-way 1-shot classification, our approach surpasses the second-best performer (ViewNet) by a margin of 1.91%. It should be noted that improvement margins are much higher on the ScanObjectNN dataset,

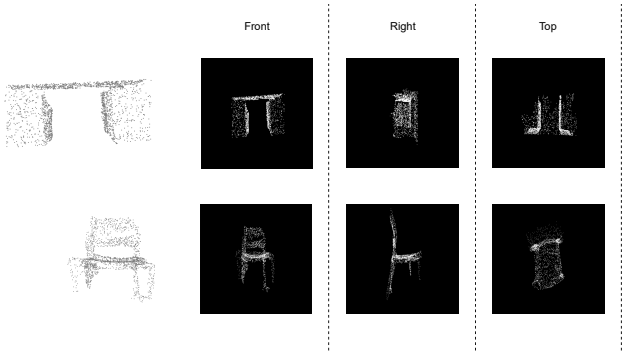

Figure 6. Example point clouds of a desk and a chair and three of the corresponding projected images for the ScanObjectNN dataset.

which contains real-world data, and is the most challenging one among the three datasets. Improvements in accuracy, and higher margins on the challenging ScanObjectNN dataset serve as the strong evidence of our model's effectiveness when dealing with real-world point cloud scans.

6. Ablation Studies

In our ablation studies, we use the ScanObjectNN dataset, known for its real-world point cloud scans, to assess the effectiveness of our method and its components. The experimental parameters mirror those outlined in Sec. 5. Our ablation studies focus on three aspects: (a) we first evaluate the effectiveness of treating three pairs of projection images individually (by sending them separately to a few-shot

head) and then incorporating a voting mechanism; (b) we study the role of using set pooling (instead of traditional average pooling and max pooling) at the end of the ResNet-18 architecture; (c) we conduct a comparative analysis of the accuracy of DepthVoting with and without data augmentation. In these ablation studies, we base our analysis on 700 meta-testing episodes, both in 5-way 1-shot and 5-way 5-shot settings.

6.1. Analysis of Voting Mechanism

In our proposed DepthVoting, we treat three pairs of projection images individually, send them separately to a few-shot head, and then adopt a voting mechanism. To analyze the effectiveness of our original method, we construct another approach, wherein features of 12 projection images (outputted by ResNet-18) are concatenated together (without being divided into sub-feature maps), and the concatenated feature is sent to the CIA few-shot head and then the classifier not using any voting. For the purposes of this ablation study, this later approach is referred to as 'Concatenated features, No Voting'. These two approaches are compared in Tab. 5. As can be seen, our DepthVoting, treating three pairs separately and incorporating a voting mechanism, provides significantly better results than 'Concatenated features, No Voting' in all folds as well as in terms of mean accuracy. More specifically, for 5-way 1-shot classification, DepthVoting achieves an impressive accuracy improvement of 9.42% over the 'Concatenated features, No Voting'. For 5-way 5-shot classification, the improvement margin is even higher at 12.18%.

		fold 0	fold 1	fold 2	Mean
5-way 1-shot	'Concat. features, No Voting'	50.06	57.14	61.77	56.32
	DepthVoting	**60.51**	**66.80**	**69.90**	**65.74**
5-way 5-shot	'Concat. features, No Voting'	61.91	68.97	71.55	67.48
	DepthVoting	**77.02**	**77.21**	**84.76**	**79.66**

Table 5. The comparison of the accuracy of our model with and without voting mechanism. **Bold** indicates the better result.

6.2. Effect of Set Pooling

We present a comparison of the performance of DepthVoting, which uses set pooling, with a model that uses the traditional ResNet-18 (i.e. set pooling is replaced by max-pooling). The results presented in Tab. 6 show that the use of set pooling enhances the model's performance, both in 1-shot and 5-shot classification tasks. In 5-way 1-shot classification, set pooling provides an accuracy improvement of 3.85% compared to using max-pooling. Similarly, in 5-way 5-shot classification, the adoption of set pooling in DepthVoting results in an accuracy increase of 6.47% compared to not using set pooling.

6.3. Effect of Data Augmentation

We perform a comparison of the performances of DepthVoting with and without data augmentation. As mentioned above, rotation is used for data augmentation. The results are summarized in Table 7. It can be observed that using rotation for data augmentation significantly improves the performance of the model, both in 1-shot and 5-shot classification tasks. In 5-way 1-shot classification, data augmentation provides an improvement of 1.32% in average accuracy compared to not using data augmentation. Similarly, in 5-way 5-shot classification, using data augmentation results in an increase of 2.25% in average accuracy compared to not using data augmentation.

		fold 0	fold 1	fold 2	Mean
5-way 1-shot	Max pooling	55.30	66.30	64.06	61.89
	DepthVoting (w/ set pooling)	**60.51**	**66.80**	**69.90**	**65.74**
5-way 5-shot	Max pooling	69.21	73.65	76.70	73.19
	DepthVoting (w/ set pooling)	**77.02**	**77.21**	**84.76**	**79.66**

Table 6. The comparison of the accuracy of the models using set pooling and the model without using set pooling. **Bold** indicates the better result.

		fold 0	fold 1	fold 2	Mean
5-way 1-shot	without data augmentation	58.21	65.26	69.78	64.42
	with data augmentation	**60.51**	**66.80**	**69.90**	**65.74**
5-way 5-shot	without data augmentation	73.46	**77.40**	81.37	77.41
	with data augmentation	**77.02**	77.21	**84.76**	**79.66**

Table 7. The comparison of the accuracy of the models with and without data augmentation. **Bold** indicates the better result.

7. Conclusion

We have proposed DepthVoting, a novel projection-based method, to perform few-shot 3D point cloud classification while increasing robustness against missing points in point clouds. DepthVoting projects the point cloud data onto six planes from different views to obtain six 2D projection images, and then performs data augmentation through 180-degree clockwise rotation of each image. Instead of extracting features from multiple projection images simultaneously, DepthVoting captures features from three pairs of projection images (obtained from opposite view angles) separately. These features are then sent to multiple few-shot heads, which share parameters. To get final predictions, DepthVoting employs a voting mechanism, aggregating information from different views. Experiments performed on the ModelNet40, ModelNet40-C, and ScanObjectNN datasets have shown that DepthVoting consistently outperforms five baseline models on 5-way 1-shot and 5-way 5-shot 3D point cloud classification. The improvements in accuracy, and higher improvement margins on the challenging ScanObjectNN dataset serve as the strong evidence of our model's effectiveness when dealing with real-world point cloud scans. Ablation studies have highlighted the important role of treating each pair of images separately and using the voting mechanism, as well as the effect of set pooling and data augmentation.

References

[1] Hanqing Chao, Yiwei He, Junping Zhang, and Jianfeng Feng. Gaitset: Regarding gait as a set for cross-view gait recognition. In *Proceedings of the AAAI conference on artificial intelligence*, pages 8126–8133, 2019. 5

[2] Jiajing Chen, Minmin Yang, and Senem Velipasalar. Viewnet: A novel projection-based backbone with view pooling for few-shot point cloud classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17652–17660, 2023. 1, 2, 3, 5, 6, 7

[3] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 264–272, 2018. 2

[4] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017. 3

[5] Ankit Goyal, Hei Law, Bowei Liu, Alejandro Newell, and Jia Deng. Revisiting point cloud shape classification with a simple and effective baseline. In *International Conference on Machine Learning*, pages 3809–3820. PMLR, 2021. 1, 3, 4

[6] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*. Lille, 2015. 3

[7] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10657–10665, 2019. 2, 6, 7

[8] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *International conference on machine learning*, pages 2554–2563. PMLR, 2017. 3

[9] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 1, 2

[10] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 1, 2

[11] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International conference on learning representations*, 2016. 3

[12] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016. 3

[13] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017. 2, 3, 6, 7

[14] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 2

[15] Jiachen Sun, Qingzhao Zhang, Bhavya Kailkhura, Zhiding Yu, Chaowei Xiao, and Z Morley Mao. Benchmarking robustness of 3d point cloud recognition against common corruptions. *arXiv preprint arXiv:2201.12296*, 2022. 2, 6

[16] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018. 2, 6, 7

[17] Sasha Targ, Diogo Almeida, and Kevin Lyman. Resnet in resnet: Generalizing residual architectures. *arXiv preprint arXiv:1603.08029*, 2016. 2, 3, 4

[18] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1588–1597, 2019. 2, 3, 6

[19] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016. 3

[20] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 1, 2

[21] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 2, 6

[22] Chuangguan Ye, Hongyuan Zhu, Yongbin Liao, Yanggang Zhang, Tao Chen, and Jiayuan Fan. What makes for effective few-shot point cloud classification? In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1829–1838, 2022. 1, 2, 3, 5, 6, 7