Why Discard if You can Recycle?: A Recycling Max Pooling Module for 3D Point Cloud Analysis

Jiajing Chen, Burak Kakillioglu, Huantao Ren, Senem Velipasalar
Syracuse University, Electrical Engineering and Computer Science Dept., Syracuse, NY, USA
{jchen152, bkakilli, hren11, svelipas}@syr.edu *

Abstract

In recent years, most 3D point cloud analysis models have focused on developing either new network architectures or more efficient modules for aggregating point features from a local neighborhood. Regardless of the network architecture or the methodology used for improved feature learning, these models share one thing, which is the use of max-pooling in the end to obtain permutation invariant features. We first show that this traditional approach causes only a fraction of 3D points contribute to the permutation-invariant features, and discards the rest of the points. In order to address this issue and improve the performance of any baseline 3D point classification or segmentation model, we propose a new module, referred to as the Recycling Max-Pooling (RMP) module, to recycle and utilize the features of some of the discarded points. We incorporate a refinement loss that uses the recycled features to refine the prediction loss obtained from the features kept by traditional max-pooling. To the best of our knowledge, this is the first work that explores recycling of still useful points that are traditionally discarded by max-pooling. We demonstrate the effectiveness of the proposed RMP module by incorporating it into several milestone baselines and state-of-the-art networks for point cloud classification and indoor semantic segmentation tasks. We show that RPM, without any bells and whistles, consistently improves the performance of all the tested networks by using the same base network implementation and hyper-parameters. The code is provided in the supplementary material.

1. Introduction

3D point cloud data analysis has a wide range of application areas, including autonomous driving, robotics, and simultaneous localization and mapping (SLAM). With the ever-increasing availability of 3D sensors, deep learning-based 3D point cloud processing has made significant strides over the past few years. However, different from 2D structured image data, 3D point cloud data is a set of unordered points, and has varying cardinality. Thus, traditional Convolution Neural Networks (CNNs) cannot be readily applied to 3D point cloud data.

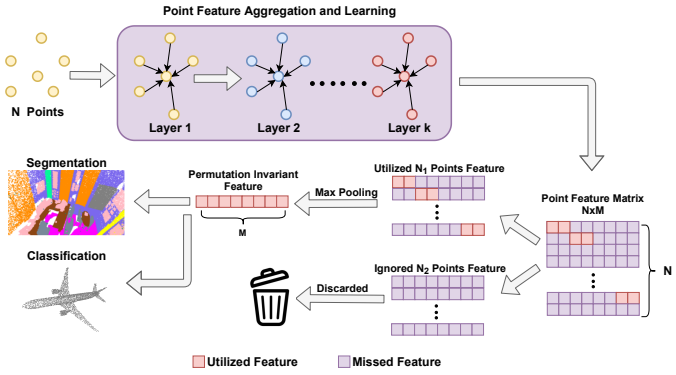


Figure 1. **Motivation.** In most point-based models, after several layers of neighbor feature aggregation and learning, an $N \times M$ feature matrix is obtained, where N is the number of points and M is each point's dimension. Max-pooling is performed at the end to obtain permutation-invariant features. Max-pooling keeps features from only part of the points (red boxes), while discarding some points' features entirely (all purple rows).

PointNet [12] is a pioneering 3D point cloud analysis work using end-to-end deep learning. It employs max-pooling operation as a symmetric function to obtain permutation-invariant features from 3D point clouds. Each point is processed independently by a shared multilayer perceptron (MLP), which does not account for local relationship of neighboring points. To remedy this, later works proposed other network structures or improved point feature aggregation approaches [13, 14, 22, 24, 27], while still employing the same max pooling operation. However, as the point feature aggregation module becomes more complex, the computational cost of the whole network increases,

*The information, data, or work presented herein was funded in part by National Science Foundation under Grant 1816732 and by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Number DE-AR0000940. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

since it needs to go through several neighbor feature aggregation layers to learn a good feature representation.

One common theme with most of the existing approaches is their use of traditional max-pooling. Fig. 1 illustrates the max-pooling operation, where red and purple boxes are the kept and discarded features, respectively. For each of the M features, the feature with the highest value among N points is kept. Thus, some points (all-purple rows) may have no contribution in this process, since none of their features are included in the final permutation-invariant feature vector. Moreover, for most point cloud classification or segmentation networks, the main computational load is due to the neighbor feature aggregation or point feature learning module. At the end, if only a small group of points contribute to the final prediction vector, and the rest of the point features are discarded, then this is also an inefficient use of the computational resources.

So, why discard if you can recycle? In order to recycle the precious set of discarded features, which are usually obtained by some complex feature aggregation and learning modules or complex network structures, we propose a novel Recycling Max-Pooling (RMP) module. The proposed RMP module performs repeated max-pooling operations among the points that were discarded by the previous max-pooling step to obtain the corresponding permutation-invariant features for training. The proposed RMP module uses the new set(s) of permutation-invariant features, obtained from the discarded points, to refine the original set of features obtained by the first max-pooling operation, and increase the performance of the original network.

Contributions. The main contributions of this work include the following:

- We first show that many baseline approaches throw away a significant portion of points after using traditional max-pooling, and this affects the model performance.
- We also show that the features thrown away indeed provide comparable performance, to the features that are kept by a baseline model, when used by themselves. Thus, it is wasteful to discard them for not only computational reasons but also for performance reasons.
- We propose a novel Recycling Max-Pooling Module (RPM) to recycle these still informative features for improved performance. To the best of our knowledge, this is the first work to explore the recycling of discarded point features, and investigate how to take advantage of a bigger portion of the point cloud.
- Our method allows refining the original permutation-invariant features only during the training process to improve a baseline network's performance.
- We provide extensive experimental results and comparisons with multiple milestone and state-of-the-art (SOTA) methods on various datasets, including ModelNet40 [23], ScanObjectNN [21] and S3DIS [1]. The results show that when the proposed RPM module is incorporated into these networks, it consistently improves the performance on point cloud classification and indoor point cloud semantic segmentation tasks.

2. Related Work

CNNs have been proved to work well on tasks involving 2D images, such as image classification [7, 18], object detection [15, 16], and semantic segmentation [10, 17]. However, different from 2D images, 3D point cloud data is unstructured, making CNNs not readily applicable to tasks involving 3D point clouds. To address this problem, PointNet [12] showed that permutation-invariant features can be obtained from 3D point clouds by a symmetric function, and used max-pooling operation for this purpose. Many following methods have been proposed based on this idea [12, 22, 24, 25], and all these methods use max-pooling at the end of their models. Similar to the survey in [26], we classify point cloud analysis methods into 3 categories: (i) multi-view based models; (ii) Volumetric models; (iii) point-based methods.

i. Multi-view-based models usually transform 3D point cloud data into 2D images by projection, and then apply existing 2D image processing models to perform prediction. A multi-view CNN (MVCNN) is proposed in [19] to perform 3D point cloud classification and segmentation by first projecting 3D point cloud into 2D images from different perspectives, and then using a CNN model to extract features from those 2D images. To address the issue of information loss, due to 3D to 2D projection, SnapNet [2] generates pairs of RGB and depth images, by taking snapshots of a point cloud, and performs point cloud semantic segmentation with the help of depth images. SimpleView [6], a more recent multi-view based method, presents that the choice of training strategy, such as learning rate decay, optimizer choice etc., has significant effect on network's performance. It first projects point cloud into six orthogonal planes to create sparse depth images, and then applies ResNet [7] to perform classification with a SOTA accuracy.

ii. Volumetric models first transform an unstructured 3D point cloud data into voxel-grids, and then use 3D CNN to perform classification and segmentation. VoxNet [11] is an earlier voxel-based model applying 3D CNN to voxelized point cloud data. However, large memory requirement and long training times are the main drawbacks of this approach. SEGCloud [20] first converts point cloud into coarse voxels, 3D fully convolutional network perform prediction on those voxels, and the prediction result will be transferred back to the raw 3D points via trilinear interpolation. PointGrid [8] is a 3D convolution-based method, which divides space into a number of grid cells. From each grid cell, a fixed number of points is selected to allow the network to learn higher-order local approximation functions. Instead of partition-

ing points into Cartesian voxels, Cylinder3D [28] partition points into cylindrical voxels. This method is more suitable for large-scale and sparsely distributed point clouds.

iii. Point-based models take raw 3D point as input directly. By performing the max-pooling operation, permutation invariant features are obtained. PointNet [8] is a pioneering work in this category. Yet, performing max-pooling among all points causes losing some local information. To address this problem, PointNet++ [13] uses a hierarchical structure. Farthest Point Sampling (FPS) is performed first to group the point cloud into local neighborhoods in each layer. Then a shared PointNet is run on each of these groups separately and aggregates all neighborhoods together for final classification and/or segmentation. DGCNN [22] proposes a dynamic edge convolution, in which the feature of every point is calculated based on its own K-nearest neighborhood that constantly changes at different layers. DPFA [3] adopts DGCNN's structure, but with an attention mechanism to aggregate neighboring points' features. GDANet [25] introduces the Geometry-Disentangle Module to dynamically disentangle point clouds into the contour and flat parts of 3D objects. In each layer, features from contour part and flat part refine all points's features for final prediction. CurveNet [24] proposes a novel point feature aggregation method, and provides SOTA performance on the ModelNet40 [23] dataset. In each layer, points learn how to make up a curve and curve's feature is learned for final prediction.

3. Recycling Max Pooling (RMP) Module

Our Recycling Max Pooling (RMP) module is designed to improve the performance of networks that use max-pooling operation during 3D point cloud processing. Instead of performing max-pooling only once, and discarding a significant portion of perfectly useful features, RMP module performs the max-pooling operation repeatedly among the points that were discarded in the previous max-pooling stage. This way permutation-invariant features are collected at multiple levels, which are then recycled to refine and improve the performance of the original model, which only uses features kept after the first max-pooling operation. The motivation behind the RMP module is provided in Sec. 3.1.

3.1. Motivation

We first perform an analysis of the percentage of points kept/utilized or discarded by several baseline models using traditional max-pooling. Our studies provide two key findings: **(i)** points that are discarded by the traditional max-pooling can provide comparable performance when used by themselves, **(ii)** a model's prediction accuracy is correlated with the point utilization percentage.

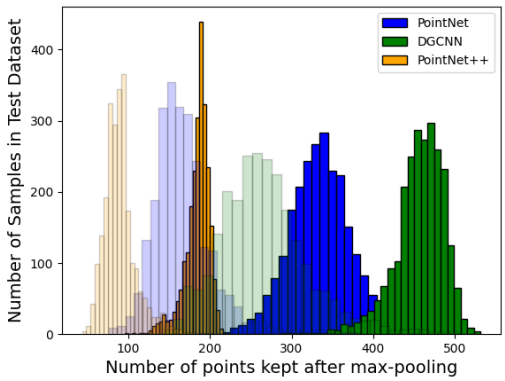


Figure 2. **Distribution of the number of points kept after one max-pooling.** Distributions for different models have distinct means. Lighter and darker shades represent the values at the beginning and end of the training, respectively.

3.1.1 Points Utilization Analysis

While existing point-based models focus on developing different point feature learning and neighbor feature aggregation modules (usually with increasing complexity), they use the same max-pooling operation in the end for permutation-invariance. As shown in Fig. 1, max-pooling operation discards features of some points entirely. To analyze the percentage of the points discarded, and its effect on the model performance, we perform experiments with PointNet, PointNet++ and DGCNN. These baselines were chosen since most point-based methods have been developed based on these three networks. After training, PointNet, PointNet++ and DGCNN are tested on the ModelNet40 classification dataset, and the number of points kept after the traditional max-pooling is recorded for each test sample. Table 1 shows the mean and standard deviation (stdev) for the number of points kept after max-pooling, and also shows the accuracy for three different models. For all the models (except PointNet++), the number of points before max-pooling is 1024. For PointNet++, a fixed number of points are sampled in each layer, thus, there are 256 points before max-pooling. Different models have different mean values, but all have small stdev values. It can also be seen that the prediction accuracy is positively correlated with the point utilization percentage, indicating that wisely recycling some of the discarded points has the promise of increasing the prediction accuracy. Fig. 2 shows the distribution of the number of points kept after max-pooling for different models, where lighter and darker shades represent the values at the beginning (when models have random weights) and end of the training, respectively. The distributions at the end of the training are the shifted versions of those before training to the right, indicating that the training process is finding various points with useful features and increasing the number of points kept after the max-pooling. This provides motivation that recycling some of these discarded points is favorable. More motivation to come in Sec. 3.1.2.

Model	No.of Pnts before Max-Pooling	Mean of no. of kept pnts	*stdev* of no. of kept pnts	% of kept pnts	Accuracy
PointNet	1024	335.8	39.6	32.80%	90.12%
PointNet++	256	184.2	15.2	72.00%	93.07%
DGCNN	1024	456.5	30.5	44.60%	92.51%

Table 1. **Point utilization analysis of different models that use traditional max-pooling**. *stdev* is the standard deviation. Models not utilizing any features from most points have lower accuracy.

We have also analyzed the number of points kept after max-pooling for each class. After recording the number of utilized points for each sample of each class, we applied a normal distribution test [4,5] on the data. For all three models, the number of points selected after max-pooling follows a normal distribution for most (32 to 33) of the 40 classes (a table is provided in the Suppl. material). We can observe that the number of points kept after max-pooling is related to the sample shape's complexity, i.e. for more complex shapes, more points are kept to represent the shape information, or vice versa. After these findings, in the next section, we analyze whether any of these discarded points and their features are useful for the task at hand.

3.1.2 Analysis of the Potential of Discarded Points

To analyze the potential of discarded points, we perform experiments by only using their permutation-invariant features (obtained via repeated applications of max-pooling) for the classification task on ModelNet40. As shown in Fig. 1, with point-based approaches, after several feature aggregation layers, a point feature matrix $\boldsymbol{P}_1^f \in \mathbb{R}^{N_1 \times M}$ is obtained, where N_1 is the number of points before the first max-pooling, and M is the feature dimension. After the first max-pooling, a permutation invariant feature vector $\boldsymbol{F}_1 \in \mathbb{R}^M$ is obtained together with the feature matrix of the discarded points $\boldsymbol{P}_2^f \in \mathbb{R}^{N_2 \times M}$, where N_2 is the number of the discarded point after the first max-pooling. Then, we apply max-pooling on $\boldsymbol{P}_2^f$ to obtain $\boldsymbol{F}_2 \in \mathbb{R}^M$. This recycling process can be repeated n times to obtain the n^{th} level permutation-invariant feature vector $\boldsymbol{F}_n \in \mathbb{R}^M$.

In order to explore the potential of permutation-invariant features from different levels, we obtain $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ for PointNet, PointNet++ and DGCNN, and use them individually to test on the ModelNet40 classification task. Accuracy values obtained with $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are provided in Table 2. For all three models, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$, by themselves, provide very similar and comparable performance to $\boldsymbol{F}_1$, yet all existing models, to our best knowledge, only make use of $\boldsymbol{F}_1$ for final prediction when these discarded points indeed have useful features that should be recycled.

Model	Using $\boldsymbol{F}_1$	Using $\boldsymbol{F}_2$	Using $\boldsymbol{F}_3$
PointNet	**90.12%**	89.64%	89.56%
PointNet++	**93.06%**	92.76%	92.92%
DGCNN	**92.51%**	92.22%	92.01%

Table 2. Classification accuracy obtained when permutation-invariant features $\boldsymbol{F}_1$,$\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are used by themselves.

3.2. Recycling Max Pooling Module

Motivated by the findings in Sec. 3.1, we propose the Recycling Max-Pooling (RMP) module to increase the percentage of utilized points by recycling the points, which are discarded by the traditional max-pooling, for training. Using the notation introduced in Sec. 3.1.2, combining $\boldsymbol{F}_1, \boldsymbol{F}_2, ..., \boldsymbol{F}_n$ simply by concatenation or addition is not the most preferable approach, which is also supported by our experiments (please see Suppl. material). Instead, our proposed RMP module first obtains $\boldsymbol{F}_1, \boldsymbol{F}_2, ..., \boldsymbol{F}_n$ by recycling the discarded points, and then refines $\boldsymbol{F}_1$ by designing a hierarchical loss function as illustrated in Fig. 3. This loss function incorporates classification loss and refinement loss, which are described below:

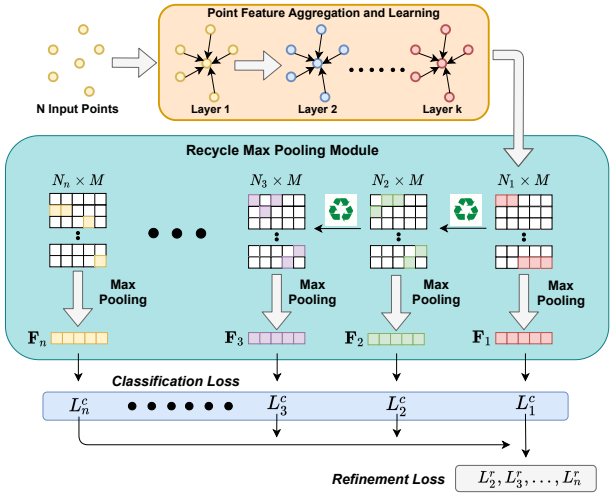


Figure 3. **Proposed RMP module.** After the point feature aggregation and learning, the classification loss and refinement loss are obtained by different level's permutation invariant feature.

3.2.1 Classification Loss

The classification losses L_1^c, L_2^c, ..., L_n^c are calculated based on the individual predictions of permutation-invariant features, $\boldsymbol{F}_1, \boldsymbol{F}_2, ..., \boldsymbol{F}_n$, respectively. L_i^c is the cross-entropy loss between $\hat{y}_i$ and y_i, where y_i is the one-hot encoded ground truth, and $\hat{y}_i$ is the soft-max prediction obtained based on $\boldsymbol{F}_i$ for $i \in \{1, ..., n\}$. Then, the classification loss is defined as:

$$L^c = \sum_{i=1}^{n} L_i^c. \tag{1}$$

Since different sets of point features are sampled at each recycling max-pooling level, and the classifier is trained on

these different batches of points, this allows the classifier to learn and generalize better.

3.2.2 Refinement Loss

In [9], an Augmenter network is used to transform the input point cloud. Then, the augmented loss, obtained by feeding the augmented data to target promotion network, is used to refine the original loss. Inspired by this, we design our refinement loss function to refine $\boldsymbol{F}_1$ by $\boldsymbol{F}_2, \boldsymbol{F}_3, ..., \boldsymbol{F}_n$. Contrary to [9], we do not employ another network to improve the target network. Instead, we only perform several layers of max-pooling, and use the permutation-invariant features $\boldsymbol{F}_i$ ($i \neq 1$), obtained at each recycling layer, to refine $\boldsymbol{F}_1$.

According to the analysis in Sec. 3.1.2 and Table 2, the accuracy obtained by only using $\boldsymbol{F}_i$, where $i \in \{2, .., n\}$, is lower than the one obtained from $\boldsymbol{F}_1$, which means $L_i^c > L_1^c$. Thus, we define the refinement loss for the recycling level i as follows:

$$L_i^r = \left| 1 - e^{(L_i^c - \rho_i L_1^c)} \right| \tag{2}$$

$$\rho_i = \alpha_i \cdot e^{(\sum_{m=1}^{k} y_m \cdot \hat{y_m})}, \tag{3}$$

where $\alpha_i > 1$, k is the number of classes, and y_m and $\hat{y_m}$ are the ground truth and the prediction based on $\boldsymbol{F}_1$, respectively. During the training process, the classification losses L_1^c, L_2^c, ..., L_n^c are minimized as described in Sec. 3.2.1. By incorporating ρ_i, Eqn. (2) promotes $L_1^c < L_i^c$ for $i \in \{2, ..., n\}$, so that best point utilization can still be achieved after the first max-pooling operation. Minimizing the L_i^r in Eqn. (2) aims to obtain $L_i^c \approx \rho_i L_1^c$ making L_i^c slightly larger (adjusted by ρ_i) than L_1^c. It is also not desirable for L_i^c to be too large, which would contradict the goal of minimizing (1). Thus, to avoid L_i^c, and in turn $L_i^c - \rho_i L_1^c$, to get too large, we take absolute value of the difference, so that L_i^c is restricted and $L_i^r >= 0$.

At the early stages of the training, the features are not yet very reliable, which might make the refinement loss not that beneficial. Therefore, we dynamically adjust the ρ_i as in Eqn. (3). When the predictions are not good, i.e. $\sum_{m=1}^{k} y_m \cdot \hat{y_m} \approx 0$, then $\rho_i = \alpha_i$. This places more attention on the predictions made based on $\boldsymbol{F}_1, \boldsymbol{F}_2, ..., \boldsymbol{F}_n$. When each level's prediction performance becomes stable, ρ_i increases and $\boldsymbol{F}_1$ starts to be refined by $\boldsymbol{F}_2, ..., \boldsymbol{F}_n$. The overall refinement loss is defined as $L^r = \sum_{i=2}^{n} L_i^r$. Combining the classification loss (L^c) and refinement loss (L^r), the final loss function is defined as:

$$L = (1 - \lambda) \cdot L^c + \lambda \cdot L^r, \tag{4}$$

where λ determines the weight of the refinement loss.

4. Experiments

In order to show that the proposed RMP module is generalizable, and can improve the performance of various networks, we have incorporated the RMP module to several milestone works and recent SOTA methods to perform point cloud classification and indoor semantic segmentation. In all the experiments (except ablation studies), we perform max-pooling twice ($n = 2$), i.e. we only use one additional recycling layer. Thus, compared to training the original network, the RMP module does not cause significant overhead.

As stated in [6], the training setup has a great impact on a network's performance. To perform fair and commensurate comparison, all the models are trained on the same machine, with the same configuration. The accuracy for the classification experiments is evaluated without any voting, i.e., $Accuracy = \frac{T}{T+F}$, where T and F are true and false classification, respectively. This is done to show the performance improvement provided solely by our proposed RMP module. Thus, an original model's accuracy reported here can be slightly different from what is reported in the corresponding papers.

During training, the accuracy values tends to fluctuate when a model's performance is close to converging. Thus, the highest recorded accuracy, from all training epochs, can not always reflect a network's learning ability. Thus, we also report the smoothed accuracy, which is computed by

$$SA_n = \beta \cdot SA_{n-1} + (1 - \beta) \cdot A_n, \tag{5}$$

where SA_n and A_n are the smoothed and highest accuracy values, respectively, at epoch n. $\beta \in [0, 1]$, and is set to be 0.99 here.

4.1. Point Cloud Classification on ScanObjectNN

In this experiment, we evaluate several milestone baselines and SOTA methods, namely CurveNet [24], DPFA [3], GDANet [25], DGCNN [22], PointNet++ [13] and PointNet [12], with and without our proposed RMP module incorporated. We perform point cloud classification on the ScanObjectNN dataset [21] without the background. This dataset contains 15k objects from 15 categories. Some example objects from this dataset are shown in Fig. 4. As can be seen, missing object parts and nonuniform distribution of points make this dataset more challenging. Table 3

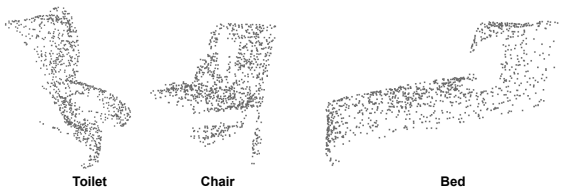


Figure 4. Example objects from the ScanObjectNN dataset.

Model Name	Highest Acc.	Smoothed Acc.
PointNet	79.39%	76.82%
PointNet++	88.17%	84.75%
DGCNN	83.10%	79.95%
GDANet	84.23%	81.29%
DPFA	84.24%	80.73%
CurveNet	83.84%	81.52%
PointNet (+RMP)	80.57% (↑1.18%)	77.43% (↑0.61%)
PointNet++ (+RMP)	**89.02%** (↑0.85%)	**85.38%** (↑0.63%)
DGCNN (+RMP)	87.0% (**↑3.97%**)	82.42% (**↑2.47%**)
GDANet (+RMP)	86.27% (↑2.04%)	82.75% (↑1.46%)
DPFA (+RMP)	85.93% (↑1.69%)	81.94% (↑1.21%)
CurveNet (+RMP)	85.54% (↑1.7%)	81.93% (↑0.41%)

Table 3. **Classification results on the ScanObjectNN dataset**. The proposed RMP module provides consistent improvement over all baselines in both the highest and smoothed accuracy values. Black bold font and blue font show the best performance and the highest increase provided, respectively.

Model Name	Highest Acc.	Smoothed Acc.
PointNet	90.12%	88.73%
PointNet++	93.06%	90.69%
DGCNN	92.51%	91.30%
GDANet	92.30%	90.59%
DPFA	93.10%	91.38%
CurveNet	92.82%	92.55%
PointNet (+RMP)	90.60% (↑0.48%)	88.74% (↑0.01%)
PointNet++ (+RMP)	93.27% (↑0.21%)	92.26% (**↑1.57%**)
DGCNN (+RMP)	93.15% (↑0.64%)	91.69% (↑0.39%)
GDANet (+RMP)	93.27% (**↑0.97%**)	91.70% (↑1.2%)
DPFA (+RMP)	**93.67%** (↑0.57%)	91.84% (↑0.46%)
CurveNet (+RMP)	93.42% (↑0.6%)	**92.98%** (↑0.43%)

Table 4. **Classification results on the ModelNet40 dataset.** The proposed RMP module provides consistent improvement over all baselines in both the highest and smoothed accuracy values. Black bold and blue fonts show the best performance and the highest increase provided, respectively.

shows the accuracy values of different SOTA networks with and without using our proposed RMP module. It can be seen that for all six models, the proposed RMP module provides consistent improvement in both the highest (as high as 3.97%) and smoothed accuracy values. Another observation is that, compared to the other five models, the RMP provides the least improvement over PointNet++. This is expected, since PointNet++ performs several samplings before the first max pooling. As shown in Table 1, PointNet++ has only 256 points before the first max-pooling operation, after which 72% of 256 points is already utilized for prediction. Thus, there is a smaller amount of points left, decreasing the amount of useful features for recycling.

4.2. Point Cloud Classification on Modelnet40

In this experiment, we evaluate the same baselines as in the above experiment with and without our proposed RMP module on the Modelnet40 [23] dataset for classification. This dataset contains 12,311 CAD models covering 40 man-made object categories, and is split into a training set containing 9843 objects, and a testing set containing 2468 objects. Some examples from this dataset are shown in Fig. 5. Table 4 shows the accuracy values of different SOTA networks with and without using our proposed RMP module. It can be seen that for all six models, the proposed RMP module provides consistent improvement in both the highest and smoothed accuracy values.

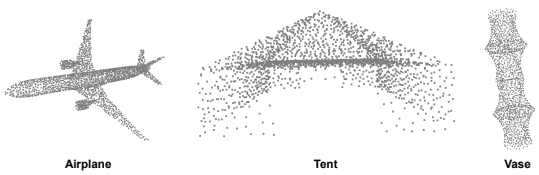


Figure 5. Example objects from the ModelNet40 dataset

Comparing Tables 3 and 4, the proposed RMP module provides more accuracy improvement over the original networks on the ScanObjectNN dataset. This can be explained by the fact that the distribution of points in the ScanObjectNN dataset are more irregular and objects have missing parts. Thus, only the features of the points, kept after one max-pooling, might not be sufficient to fully represent object shape information. By learning from permutation-invariant features obtained at different layers, a more complete shape information is obtained.

4.3. Semantic Segmentation on S3DIS

Among the six models tested for the classification task on the ScanObjectNN and Modelnet40 datasets, PointNet, DGCNN and DPFA offer max-pooling based semantic segmentation network structures in their papers. Thus, we also integrated our RMP module on these three models to evaluate it on a different task of semantic segmentation. PointNet++ also has the semantic segmentation structure, but its segmentation model is based on interpolation and upsampling rather than max-pooling. Thus, PointNet++ is not included in the semantic segmentation experiments.

S3DIS dataset [1] is a large indoor point cloud dataset. It contains 6 areas covering 271 rooms. Each point belongs to one of 13 classes: {clutter, ceiling, floor, wall, beam, column, door, window, table, chair, sofa, bookcase, board}. Following PointNet [12], we divide each room into blocks of $1m \times 1m \times z$, where z is the room height in meters. 4096 points are randomly selected from each block as the input to the network. For these 6 areas, 6-fold cross validation is performed for all the models with and without our RMP module. At each fold, one area is set aside for testing, and the models are trained on the remaining areas. The results are summarized in Table 5. For all of the 6 areas, both the overall accuracy (OA) and mIoU of all three models are increased. The average of all 6 folds is also reported showing

both OA and mIoU are improved for all 3 models. Example outputs for qualitative comparison are shown in Fig. 6.

Testing Area	Model	OA	MIoU
Area1	PointNet	77.31%	51.60%
	DGCNN	82.10%	62.94%
	DPFA	90.32%	70.29%
	PointNet(+RMP)	82.84% (↑**5.53%**)	59.22% (↑**7.62%**)
	DGCNN(+RMP)	85.62% (↑3.52%)	63.78% (↑0.84%)
	DPFA(+RMP)	**90.63%** (↑0.31%)	**71.75%** (↑1.46%)
Area2	PointNet	72.33%	34.96%
	DGCNN	78.24%	36.95%
	DPFA	87.71%	53.95%
	PointNet(+RMP)	78.62% (↑**6.29%**)	39.7% (↑4.74%)
	DGCNN(+RMP)	79.88% (↑1.64%)	43.02% (↑**6.07%**)
	DPFA(+RMP)	**90.63%** (↑2.92%)	**54.93%** (↑0.98%)
Area3	PointNet	83.47%	47.46%
	DGCNN	88.74%	62.03%
	DPFA	90.45%	66.04%
	PointNet(+RMP)	84.91% (↑**1.44%**)	57.57% (↑**10.11%**)
	DGCNN(+RMP)	89.7% (↑0.96%)	**70.23%** (↑8.2%)
	DPFA(+RMP)	**90.58%** (↑0.13%)	66.14% (↑0.1%)
Area4	PointNet	73.05%	35.71%
	DGCNN	80.62%	42.65%
	DPFA	85.94%	**51.08%**
	PointNet(+RMP)	76.93% (↑**3.88%**)	41.25% (↑**5.54%**)
	DGCNN(+RMP)	82.51% (↑1.89%)	45.72% (↑3.07%)
	DPFA(+RMP)	**87.81%** (↑1.87%)	53.87% (↑2.79%)
Area5	PointNet	78.76%	42.00%
	DGCNN	82.60%	46.97%
	DPFA	87.47%	52.96%
	PointNet(+RMP)	79.05% (↑0.29%)	43.23% (↑1.23%)
	DGCNN(+RMP)	84.25% (↑**1.65%**)	48.54% (↑1.57%)
	DPFA(+RMP)	**88.17%** (↑0.7%)	**54.58%** (↑**1.62%**)
Area6	PointNet	83.81%	57.86%
	DGCNN	84.53%	64.57%
	DPFA	92.20%	75.23%
	PointNet(+RMP)	85.72% (↑1.91%)	63.9% (↑6.04%)
	DGCNN(+RMP)	87.94% (↑**3.41%**)	71.89% (↑**7.32%**)
	DPFA(+RMP)	**92.34%** (↑0.14%)	**77.33%** (↑2.1%)
6-fold Cross Validation Average	PointNet	78.12%	44.94%
	DGCNN	83.16%	52.68%
	DPFA	89.02%	61.59%
	PointNet(+RMP)	81.35% (↑**3.23%**)	50.81% (↑**5.87%**)
	DGCNN(+RMP)	84.39% (↑1.23%)	57.2% (↑4.52%)
	DPFA(+RMP)	**89.75%** (↑0.73%)	**63.17%** (↑1.58%)

Table 5. **Segmentation results on the S3DIS dataset** with 6-fold cross validation. Overall, the proposed RMP module increases both the Overall Accuracy (OA) and mean Intersection over Union (mIoU) for all models.

5. Ablation Study

Since recent works GDANet and DPFA both adopt the feature-wise neighbor searching strategy proposed in DGCNN, we used DGCNN as the baseline network to perform the ablation studies. Since the benefits of the proposed RMP are more evident on irregular and/or incomplete point sets (as discussed in Sec. 4.1), ScanObjectNN datasets is used for the ablation studies.

5.1. Analysis of n, the number of Max-Poolings

We studied the effect of the number of times max-pooling operation is repeated. For this experiment, $\alpha = 1.7$ and $w = 0.5$. When $n = 1$, it reduces to the original network. Fig. 7(b), shows the plot of the smoothed and highest accuracy values versus n. The best smoothed and highest accuracy values are obtained when $n = 2$ and $n = 3$, respectively. When n is increased further the performance degrades. This makes sense since the first 2 or 3 max-pooling levels pick up most of the points with useful features, capturing an object's shape etc., and the remaining points might contain more noise than useful information. Fig. 7 (a) shows the smoothed accuracy plot of DGCNN, incorporating the RMP module, versus the training epochs, for different numbers of max-pooling operation. The curves for $n = 2$ and $n = 3$ almost entirely overlap, and provide the highest accuracy.

5.2. Analysis of weight value λ

In Eqn. (4), λ is the weight of the refinement loss determining its contribution to the overall loss function. We analyze the effect of different values of λ on the performance when $n = 2$ and $\alpha = 2.1$. The results are shown in Table 6. As can be seen, as the value of λ increases from 0.5 to 0.8, the accuracy values also increase in general. The accuracy improvement ranges between 0.85% and 3.97%, and the highest performance increase is obtained when $\lambda = 0.8$. When $\lambda > 0.8$, the performance does not increase anymore. This makes sense, since when $\lambda = 1$, e.g., the overall loss is equal to the refinement loss, i.e. the classification loss is not used. This is not the right approach, since it does not make sense to use the refinement loss if each $\boldsymbol{F}_i$ is not equipped with the prediction ability (also supported by the results in the table).

Model	**λ**	**Highest Acc.**	**Smoothed Acc.**
DGCNN	None	83.10%	79.95%
DGCNN (+RMP)	0.5	83.95%(↑0.85%)	81.32%(↑1.37%)
DGCNN (+RMP)	0.6	85.14%(2.04%)	81.79%(↑1.84%)
DGCNN (+RMP)	0.7	85.03%((↑1.93%))	81.82%(↑1.87%)
DGCNN (+RMP)	0.8	**87.07%(↑3.97%)**	**82.42%(↑2.47%)**
DGCNN (+RMP)	0.9	84.69%(↑1.59%)	81.54%(↑1.59%)
DGCNN (+RMP)	1	23.44%(↓59.66%)	25.48%(↓54.47%)

Table 6. Analysis of different λ values on the accuracy

5.3. Analysis of hyper-parameter α

In Eqn. (2), α_i is a constant, which is used to compute the refinement loss of L_i^r, which is employed for $\boldsymbol{F_i}$ to refine $\boldsymbol{F_1}$. For this ablation study, we only perform recycling once, i.e. max-pooling is performed a total of 2 times ($n = 2$). Thus, we only have α_2, which is referred to as α here for simplicity. The value of λ is set to 0.5. We analyze the effect of different α values on accuracy, and the results are shown in Table 7. When α is 1.2, 1.5 or 1.8, the improvement provided over DGCNN ranges between 1.02% and 1.7%. $\alpha = 1.5$ provides the highest performance increase for both the highest (1.7%)and smoothed accuracy (1.87%). When α is 2.1 the improvement provided over DGCNN is slightly less (0.85%) compared to other α values.

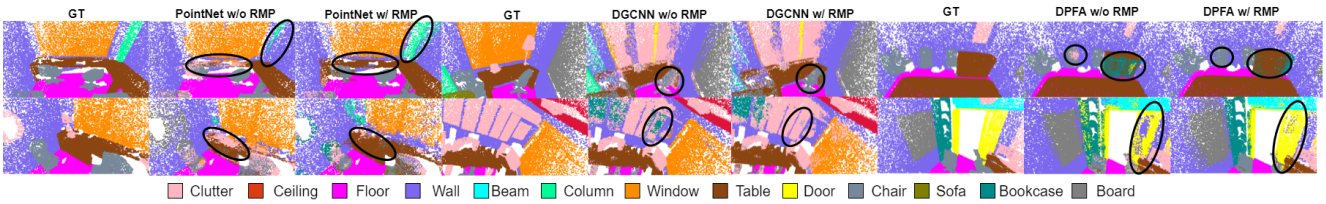
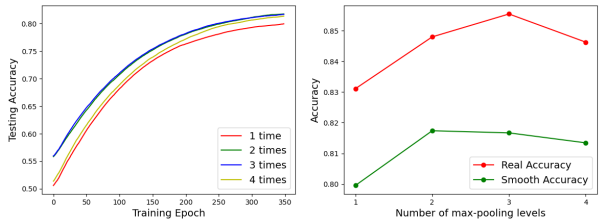


Figure 6. **Example segmentation outputs for qualitative comparison**. Outputs of PointNet, DGCNN and DPFA without (w/o) and with (w/) incorporating our proposed RMP module. Some regions are marked by black ellipses to show the improvement provided by RMP.



(a) (b)

Figure 7. **Analysis of number of max-pooling levels** (n) (a) Smoothed accuracy plot during training for different values of n, (b) the smoothed and highest accuracy values versus n.

Model	α	Highest Acc.	Smoothed Acc.
DGCNN	None	83.10%	79.95%
DGCNN(+RMP)	1.2	84.12%(↑1.02%)	81.09%(↑1.14%)
DGCNN(+RMP)	1.5	**84.80%(↑1.70%)**	**81.82%(↑1.87%)**
DGCNN(+RMP)	1.8	84.46%(↑1.36%)	81.75%(↑1.80%)
DGCNN(+RMP)	2.1	83.95%(↑0.85%)	81.32%(↑1.37%)

Table 7. Analysis of different α values on the accuracy.

5.4. Analysis of the number of input points

In order to analyze the performance with varying input point sparsity, we evaluated DGCNN, with and without the proposed RMP module, by using 512, 1024 and 2048 points, in turn, as input. The results are shown in Table 8. With the proposed RMP module, the accuracy obtained with 512 points (when input data is sparse) is almost the same as the one with 1024 points (0.78% difference), since the RMP module allows making use of more points via recycling. RMP module improves the performance of DGCNN by 2.04% even when only 512 input points are used. Over different number of input points, the RMP module provides a stable performance increase.

Model	No. of Inp. Pnts	Highest Acc.	Smoothed Acc.
DGCNN	512	81.97%	78.96%
DGCNN	1024	83.10%	79.95%
DGCNN	2048	84.18%	81.19%
DGCNN(+RMP)	512	84.01%(↑2.04%)	80.86%(↑1.9%)
DGCNN(+RMP)	1024	84.79%(↑1.69%)	81.74%(↑1.79%)
DGCNN(+RMP)	2048	86.73%(↑2.55%)	83.15%(↑1.96%)

Table 8. Analysis of performance with different number of input points

5.5. Analysis of Training Time

The training times of different models, with and without the proposed RMP module, on the ScanObjectNN dataset are listed in Table 9. Original GDANet and CurveNet provide very slight increase (1.3% and 1.57%, resp.) in the smoothed accuracy compared to DGCNN, while requiring much longer training time (5.5 to 6.5 fold increase) due to their complex feature aggregation layers. In contrast, our proposed RMP module increases the highest accuracy of DGCNN by 3.97%, with only a small overhead on training time (10.97s or 1.46 fold increase per epoch). Since the recycling is only performed during training, our approach does not affect the inference times.

Model	Training time (s)	Highest Acc.	Smoothed Acc.
DGCNN	23.53	83.10%	79.95%
DGCNN (+RMP)	34.50	87.07% (↑3.97%)	82.42(↑2.47%)
GDANet	129.41	84.23%	81.29%
GDANet (+RMP)	137.98	86.27% (↑2.04%)	82.75%(↑1.46%)
CurveNet	152.77	83.84%	81.52%
CurveNet (+RMP)	158.09	85.54%(↑1.7%)	81.93%(↑0.41%)

Table 9. Per epoch training time and accuracy of different models on the ScanObjectNN dataset

6. Discussion and Conclusion

Max-pooling is a commonly used approach to obtain permutation-invariant features for point cloud processing tasks. In this paper, we have first shown that methods using traditional max-pooling throw away a significant portion of points and that the features of these discarded points indeed provide comparable performance, to the features that are initially kept, when used by themselves. Thus, it is wasteful to discard them for not only computational reasons but also for performance reasons. To address this, we have proposed a novel Recycling Max-Pooling Module (RMP) to recycle these still informative features for improved performance. We have presented the refinement loss that, when combined with the classification loss, allows recycled features to refine the initially kept features. We have performed extensive experiments on the ModelNet40, ScanObjectNN, and S3DIS datasets for classification and segmentation tasks, and shown that the proposed RMP module is generalizable to various networks, and consistently improves the performance of several SOTA baselines. Since the proposed approach goes through an extra level to recycle the discarded features during training, this performance improvement comes with a slight increase in the training time.

References

[1] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. 3d semantic parsing of large-scale indoor spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1534–1543, 2016. 2, 6

[2] A. Boulch, B. Le Saux, and N. Audebert. Unstructured point cloud semantic labeling using deep segmentation networks. 3DOR@ Eurographics, 3, 2017. 2

[3] J. Chen, B. Kakillioglu, and S. Velipasalar. Background-aware 3d point cloud segmentation with dynamic point feature aggregation. arXiv:2111.07248, 2021. 3, 5

[4] R. D'AGOSTINO and E. S. Pearson. Tests for departure from normality. empirical results for the distributions of b 2 and b. Biometrika, 60(3):613–622, 1973. 4

[5] R. B. d'Agostino. An omnibus test of normality for moderate and large size samples. Biometrika, 58(2):341–348, 1971. 4

[6] A. Goyal, H. Law, B. Liu, A. Newell, and J. Deng. Revisiting point cloud shape classification with a simple and effective baseline. International Conference on Machine Learning, 2021. 2, 5

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016. 2

[8] T. Le and Y. Duan. Pointgrid: A deep network for 3d shape understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 9204–9214, 2018. 2, 3

[9] R. Li, X. Li, P.-A. Heng, and C.-W. Fu. Pointaugment: an auto-augmentation framework for point cloud classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6378–6387, 2020. 5

[10] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3431–3440, 2015. 2

[11] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 922–928. IEEE, 2015. 2

[12] C. R. Qi, H. Su, K. Mo, and L. 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, 5, 6

[13] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. arXiv preprint arXiv:1706.02413, 2017. 1, 3, 5

[14] S. Qiu, S. Anwar, and N. Barnes. Geometric back-projection network for point cloud classification. IEEE Transactions on Multimedia, 2021. 1

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 779–788, 2016. 2

[16] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28:91–99, 2015. 2

[17] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, pages 234–241. Springer, 2015. 2

[18] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014. 2

[19] H. Su, S. Maji, E. Kalogerakis, and E. 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

[20] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese. Segcloud: Semantic segmentation of 3d point clouds. In 2017 international conference on 3D vision (3DV), pages 537–547. IEEE, 2017. 2

[21] M. A. Uy, Q.-H. Pham, B.-S. Hua, D. T. Nguyen, and S.-K. Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In International Conference on Computer Vision (ICCV), 2019. 2, 5

[22] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. Acm Transactions On Graphics (tog), 38(5):1–12, 2019. 1, 2, 3, 5

[23] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. 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, 3, 6

[24] T. Xiang, C. Zhang, Y. Song, J. Yu, and W. Cai. Walk in the cloud: Learning curves for point clouds shape analysis. arXiv preprint arXiv:2105.01288, 2021. 1, 2, 3, 5

[25] M. Xu, J. Zhang, Z. Zhou, M. Xu, X. Qi, and Y. Qiao. Learning geometry-disentangled representation for complementary understanding of 3d object point cloud. arXiv preprint arXiv:2012.10921, 2020. 2, 3, 5

[26] J. Zhang, X. Zhao, Z. Chen, and Z. Lu. A review of deep learning-based semantic segmentation for point cloud. IEEE Access, 7:179118–179133, 2019. 2

[27] Z. Zhang, B.-S. Hua, and S.-K. Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1607–1616, 2019. 1

[28] X. Zhu, H. Zhou, T. Wang, F. Hong, Y. Ma, W. Li, H. Li, and D. Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9939–9948, 2021. 3