VARIATIONS ON S-NET FOR CLASSIFICATION

Yael Bekerman
Tel-Aviv University
ID: 032585721
yael1204@gmail.com

Roy Shen-Tzur
Tel-Aviv University
ID: 208712695
royst10@gmail.com

March 7, 2019

ABSTRACT

The use of point clouds has been increasing over recent years, with applications ranging from gaming to autonomous cars. Processing large point clouds with deep neural networks uses high computational power, has high costs and takes a lot of time. Therefore, sampling those point clouds is highly necessary. A popular sampling technique is Farthest Point Sampling (FPS) which isn't task related. Dovrat et al. showed that it is better to learn how to sample and proposed a new deep learning architecture that aims to simplify 3D point clouds, termed S-NET.

In our work, we explored and implemented several variations on their network for classification purposes. We show that using an unsupervised approach for classification is not only possible but even preferable and improves the original S-NET results. We also show that it is possible to obtain even better results by training S-NET only with samples that the classifier have high confidence in its labeling. We also investigated if it's possible to work with smaller and different datasets, that were not tested in the original paper.

1 Introduction

With increasing popularity of point cloud representations in different fields, including real time systems, the ability to sample point clouds has become a necessity. The question of how to sample is not new, and solutions such as furthest point sampling has been suggested and used before. Dovrat et al. [1] presented a new, deep learning based sampling approach, which is based on the task network itself (in our case a classification task network). The suggested architecture, called S-NET is based on the work of PointNet, and can process unordered point clouds. During the training process, the network tries to optimize two objectives. First, similarity to the original shape, and second, optimal sampling for the task network. This leads to two loss terms: a sampling loss and a task loss. Their work showed better results then non-learning based sampling methods.

We were inspired by their approach and tried to explore it furthermore while focusing on the classification task. We noticed that throughout the training process, only labeled data was used, while also using the ground-truth labels when calculating the task loss. We wished to exploit the classifier's abilities by using it's prediction instead of the ground-truth labeling. By doing so, we didn't only manage to use S-NET in an unsupervised way, but also to improve the accuracy. We also show that we can improve this result by training only with samples the classifier has high confidence in. We set different thresholds and tested the effect on the resulted accuracy.

We also wanted to show that it's possible to train the sampler with different datasets and checked if it can work with a small amount of data. We saw that for small sampling ratios, a small dataset can give good results, but can decrease the classifier's accuracy for bigger sampling ratios. The importance of all of our tested variations on S-NET is to show that if one is acquired a good classification trained task network, it is not necessary to acquire expensive, large labeled dataset as well.

2 Related Work

Sampling methods: Our project was based on Dovrat et al. paper: "Learning to Sample" [1]. They used a learned task-specific sampling approach for point clouds. Given a fixed pre-trained task network (e.g. – classification network), S-NET generates a smaller point cloud that is optimized to the task itself. Since straightforward sampling is not differentiable, S-NET does not output a subset of the original point cloud and uses a sampling loss to achieve similarity to the original shape. Therefore, a post-processing step is performed at inference time, matching the produced points with real points from the original point cloud. An illustration of this network is demonstrated in {Figure 1}

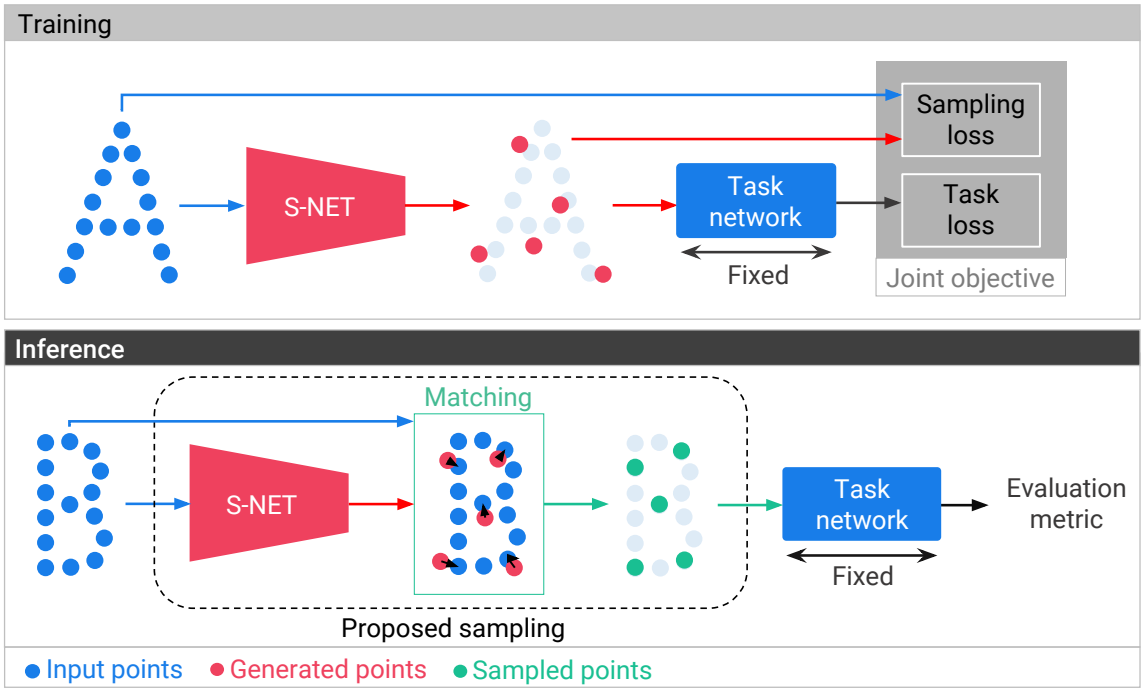

Figure 1: **An illustration of Dovrat et al. learned sampling approach.**

The architecture of S-NET is based on that of PointNet, with some modifications. First, input points undergo a set of 1x1 convolution layers (joint weights), resulting in a per point feature vector. Then, a symmetric feature-wise max pooling operation (permutation invariance) is used to obtain a global feature vector. Afterward comes several fully connected layers. The output of the last layer is the set of generated points. An illustration of S-NET architecture is demonstrated in {Figure 2}

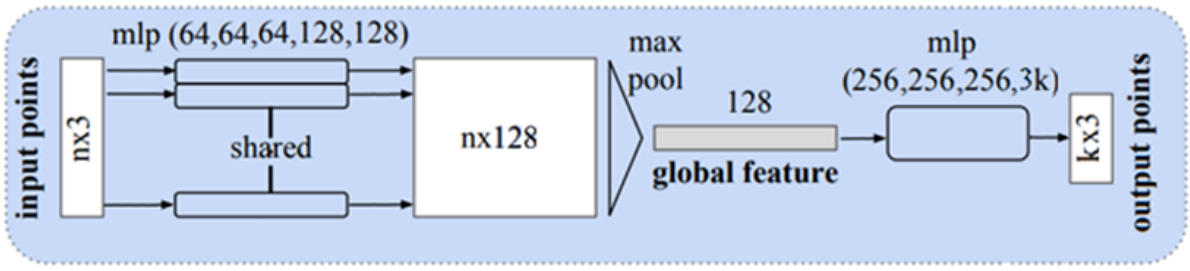

Figure 2: **S-NET architecture.**

A lot of sampling techniques were available before Dovrat et al. work, however they did not consider the objective of the task itself and are not learned methods. The most commonly used approach is Farthest Point Sampling (FPS) [2, 3].

This method takes into account the structure of the point cloud and selects a group of points that are farthest apart from each other. This method optimizes a geometric error (coverage of the shape) but oblivious to the task

Classification methods: In recent years, deep learning techniques have been applied to point cloud classification. We used PointNet, the work of Qi et al [4]. PointNet was the first neural network that operates directly on unordered point cloud data. They constructed their network from per-point multi-layer perceptrons, a symmetric pooling operation and several fully connected layers. It can accept varied number of input points, due to convolution max-pool operation (indifferent to the number of points). PointNet used FPS to reduce the number of input points. Later on, they extended their network architecture for hierarchical feature learning. [5]

3 Data

The dataset we were working with for our project is ModelNet40 [6]. It is labeled point clouds of 3D mesh models from 40 categories, mostly made by 3D modeling tools $Figure$ 3. ModelNet40 is composed of 12311 samples, splits 9843 for training and 2468 for testing. Each sample is 2048 points represented in (x,y,z) vector. Our preproccesing of the data was similar to S-NET preprocessing, the data was rotated along the up-axis and gaussian noise was added. For smaller dataset we took only 20% of the training samples and for unsupervised approach we used only the point cloud itself for the training, without using the labels.

Figure 3: **ModelNet dataset.**

For trying to train the sample network on a different dataset then the classification network we used 3D-MNIST dataset. This dataset contains 3D labeled point clouds generated from the original images of the MNIST dataset of 10 categories. It is composed of 12000 samples, splits 10000 for training and 2000 for testing. Each sample is 256 points represented in (x,y,z) vector. Here we also tried training with different amount of data, and also tested the same unsupervised approach as before.

4 Methods

We tested and implemented a few variations on S-NET for classification, that we'll discuss now in great detail:

Unsupervised S-NET training: We chose an unsupervised approach when training S-NET since using unlabeled dataset has a lot of benefits. Firstly, labeled datasets are not always available and might be more expensive to obtain. Secondly, it's not always possible to retrain the classification network, since only the model is provided and not the code itself.

Our implementation works on unlabeled data and uses a pre-trained PointNet as our fixed task network. We use the original full point cloud to find the approximated labels by feeding it through the given trained classification network and using simple softmax and one-hot techniques we had learn in class. We changed the task loss to use the classifier's decision instead of the ground truth label and trained the sampler on the desired number of points. That way, S-NET is not trying to impose a label different then the classifier's decision, but tries to preserve the features that the classifier detected in the original point cloud. Since we need to use the classifier model twice in the same tensorflow graph, each time with different number of points, we had to use different variables scopes.

An illustration of the proposed unsupervised sampling approach is shown in {Figure 4}

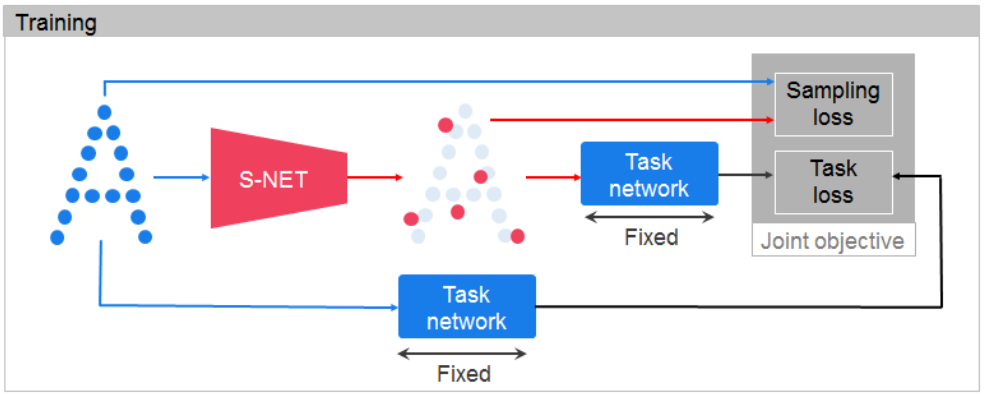

Figure 4: **An illustration of the proposed unsupervised sampling approach.**

Unsupervised S-NET training using label-confidence threshold: We wish to expend the idea of training the sampling network according to classification label and not the ground-truth label to achieve better results. This time, we trained our model only with samples that the classification task network had high confidence in. We performed softmax on the last layer to get the percentages of confidence in each label. We looked at the maximal value as the classifier's confidence and compared it to the threshold, ignoring samples that didn't have high enough confidence.

S-NET with smaller dataset and different dataset: Large datasets aren't always available and sometimes we have a trained model of the classifier task but don't have the dataset it was trained on. That is why we tried to explore if it is possible to use a trained fixed classifier and use a small dataset when training S-NET. First, we used the ModelNet40 dataset and tried to use only 20% of the training data and checked how it affects the sampler results. We saw that when the number of sampled points is low, the accuracy decreases, but for a large number of sampled points, the accuracy degradation is small. Next, we tried to train the sampler with the 3D MNIST dataset and tested the accuracy degradation as a funciton of the training dataset size, for a certain sampling ratio.

5 Experiments

5.1 Unsupervised S-NET Training

We tested our unsupervised approach on various number of points and on two different datasets: ModelNet40 and 3D-MNIST. In order to test our new methods, first we had to train two classifiers, one for each dataset on the full point cloud. Then, we trained S-NET as proposed in the original paper so we can compare it to our new method. Our implementation worked on unlabeled data and used full point cloud pre-trained PointNet as our fixed task network. We used the original full point cloud to find the approximated labels by feeding it through the given trained classification network.

It's clear that our unsupervised approach outperforms the original paper's results. {Figure 5} and {Figure 6} shows the classification accuracy against sampling ratio for both supervised and unsupervised learning for each of the dataset. Our new method has equal or better accuracy for all sampling ratios when using ModelNet40, and is outperformed by the original implementation only for one sampling ratio when using 3D-MNIST. We noticed that our approach is especially better when the number of sampled points is small. The difference gets up to 3%.

5.2 Unsupervised S-NET training using label-confidence threshold

We wanted to see if it's possible to improve the accuracy results by choosing only samples the classifier had high confidence in but without losing big portion of our dataset. In order to see what is the best threshold to use we printed out histogram of the confidence level of the classifier. For ModelNet40 we saw that a threshold of 0.8 dropped around 10% of the training data. We also tested two different threshold values, 0.7 and 0.9, on the 3D MNIST dataset to test how different threshold values affect different sampling ratios. We saw that using the correct threshold per number of sampling points improves the results. Even though large threshold value means less data, we can see that for small sampling ratios (large amount of sampled points), large values can lead to better results. When sampling only a small

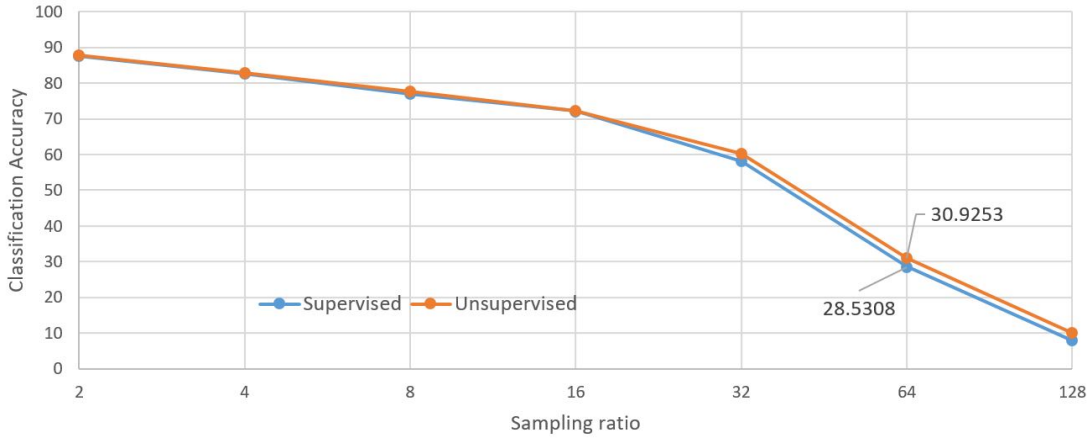

Figure 5: **ModelNet40:** Supervised S-NET Vs. Unsupervised S-NET.

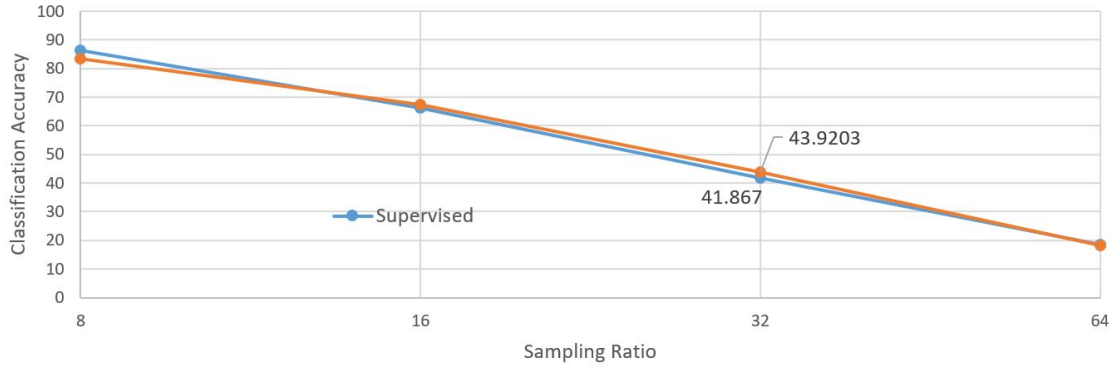

Figure 6: **3D-MNIST:** Supervised S-NET Vs. Unsupervised S-NET.

amount of points, we see that the lack of data decreases the accuracy and the threshold should be decreased to obtain good accuracy. However, we chose a global threshold and maybe we can improve the results even better if we would choose different threshold for each label and different threshold for different number of sampled points.

ModelNet40 3D-MNIST

# of points	SNET	SNET Unsupervised	SNET Unsupervised Threashold=0.8
8	7.83	9.98	9.21
16	28.53	30.93	28.25
32	58.08	60.15	**60.88**
64	72.16	72.20	**72.61**
128	76.91	77.64	77.64
256	82.71	82.87	**84.01**
512	87.70	87.95	87.74

Figure 7: **ModelNet40 Table Summary.**

# of points	SNET	SNET Unsupervised	SNET Unsupervised Threashold=0.7	SNET Unsupervised Threashold=0.9
4	18.46	18.21	18.22	17.34
8	41.87	43.92	**46.25**	38.52
16	66.14	67.34	64.35	67.07
32	86.44	83.54	85.65	**87.09**

Figure 8: **3D-MNIST Table Summary.**

We can see in ModelNet40 {Figure 7} that for 32, 64 and 256 sampled points we were able to improve the unsupervised accuracy in 0.5-1% more. Similar results are available in 3D-MNIST {Figure 8} for 8 and 32 sampled points with different thresholds.

5.3 S-NET with smaller dataset

In this experiment we wished to see if it is possible to train the sampler on a smaller dataset. We took only 20% of ModelNet40 dataaset {Figure 9} and 10% of 3D-MNIST dataset{Figure 10}. We can see that when the number of points is smaller the loss of data is more crucial.

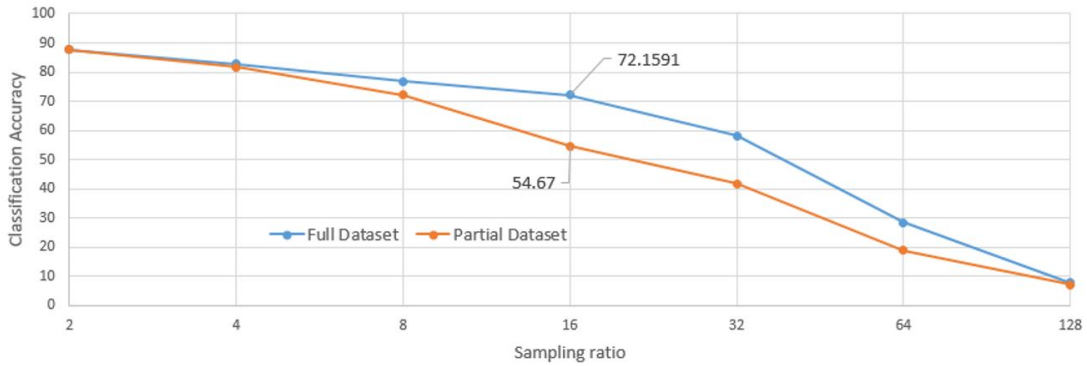

Figure 9: **ModelNet40:** Full dataset Vs. 20% of the dataset.

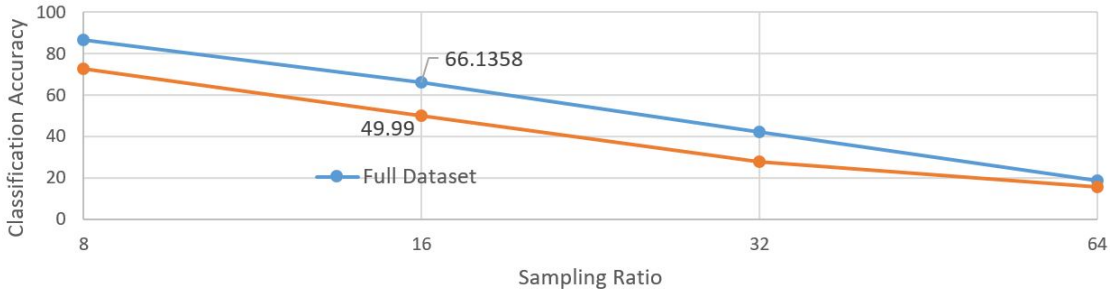

Figure 10: **3D-MNIST:** Full dataset Vs. 10% of the dataset.

We also checked how different number of training samples affect the accuracy and the resutls are provided in {Figure 11} When S-NET was trained with the full 10000 samples it got to 66.14% accuracy. We can see that with half the samples we get a very similar resutls.

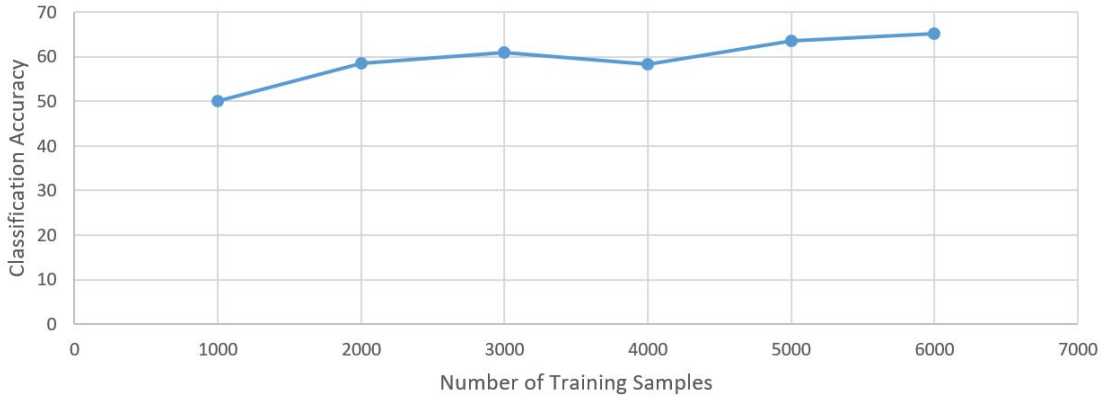

Figure 11: **3D-MNIST:** Different portions of data on 16 sampled points.

6 Conclusion

We chose Dovrat et al. paper that presented a method that learns how to sample point clouds for a specific task, which outperforms the FPS method. We tested a few variations on this network, using PointNet classification network as the task network. Our results are summarized below:

- We used an unsupervised approach and improved S-NET performance. We showed that training the sampler using the classifier's decision instead of the ground truth label can improve the accuracy results. That is because when we used the classifier's decision, S-NET tried to keep the features it detected in the original point cloud.
- We showed that S-NET can be used and give good results on complicated classification tasks even if you don't have the same expensive and labeled dataset that was used to train the classifier.
- We demonstrate that it is possible to achieve better results if you train S-NET only with dataset samples that the classifier has high confidence in, and choose the threshold according to the sampling ratio. Higher sampling ratio means less data, which means that a smaller threshold is probably better. In a similar manner, when sampling with a low sampling ratio, a high threshold value can lead to better results since small amount of training samples doesn't decrease accuracy for large amount of sampled points.
- We tried to use smaller dataset for training S-NET and showed that it is possible to use a small dataset and achieve good results only if the number of sample points is close to the original one.

This project gave us a small taste of the point cloud world and its complexity. We saw that unlike in 2D images which has a lot of application, the 3D representation is still unclear and there's a lot of work to be made. Also, this project gave us new technical skills of compiling and using c++ files in python environment and working in tensorflow with a few networks on one graph. For future work we would advise to define a new loss based on a few different classification networks. The sampling network can then be trained with multiple task networks at ones and output a point cloud that is robust and gives good results to any new classification task. Another idea is to create a combined loss for classification network and reconstruction network and train a sampler that outputs points good enough for both tasks. (We actually tried to implement one of these ideas but had a problem with one of the task's loss and couldn't fix it in time for the deadline).

References

[1] O. Dovrat, I. Lang, and S. Avidan, "Learning to sample," 2018.

[2] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, "The Farthest Point Strategy for Progressive Image Sampling," *IEEE Transactions on Image Processing*, vol. 6, pp. 1305–1315, 1997.

[3] C. Moenning and N. A. Dodgson, "Fast Marching farthest point sampling," *Eurographics Poster Presentation*, 2003.

[4] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 652–660, 2017.

[5] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2017.

[6] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A Deep Representation for Volumetric Shapes," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1912–1920, 2015.