Self-Supervised Learning in 3D Point Clouds

Likhit Nayak
Georgia Institute of Technology
Digital Image Processing Team
Atlanta, USA
likhitnayak@gmail.com

Soo Kwon Ha
Georgia Institute of Technology
Digital Image Processing Team
Atlanta, USA
sookwon36@gmail.com

Zhongyou Chen
Georgia Institute of Technology
Digital Image Processing Team
Atlanta, USA
rightchen18@gmail.com

1. Introduction

The Machine Vision System (MVS) has become progressively better in the task of understanding our real world due to the rise of powerful computational resources such as GPUs and the availability of a large amount of data. [1] However, the machine's understanding of the real world is based largely upon 2D images, whereas our Human Vision System (HVS) is able to process and analyze 3D information. In order for the MVS to emulate the HVS, it needs to understand and classify 3D scenes as well. With the recent developments in depth sensing technology, there is a large collection of three-dimensional datasets, and a large percentage of these datasets are unlabelled.

1.1 Point Clouds

In this paper we utilize deep learning architectures capable of reasoning about point cloud models. Point clouds provide a natural and flexible representation of understanding objects in the MVS. These representations are easily captured by modern scanning devices and techniques for applications in prototyping, medical imaging, robotics, self-driving cars. [2] Point clouds are easier to learn from given that they do not possess the irregularities and complexities of meshes or 3D voxels. Voxels and image grids result in voluminous data and present complications that mask invariances in the data models. For this reason, we choose point clouds as our input representation for 3D geometry, and the PointNet architecture, which caters the properties of point clouds. [3]

1.2 Self-Supervised Learning

Self-supervised learning minimizes the amount of human supervision needed for learning a task. The main benefit of self-supervised learning is the reduction of time and monetary cost associated with annotation. [4] Some of the other benefits include learning rare observations, minimization of annotation errors, and the avoidance of proprietary and privacy concerns. Techniques in the self-supervised domain present superior performances on many challenging benchmarks and have been used to learn useful representations from unlabelled datasets of 2D images. [5] The learned representations have resulted from training a network to solve "pretext" tasks which are auxiliary tasks that predict one part of the data from another. A pretext task does not require labels annotated by humans. Instead, it predicts pseudo-labels computed by the raw data itself. An example of a pretext task for 2D images consists of using the grayscale version of the training set and using the colors as pseudo-labels. [6] Another task consists of segmenting a 2D image into 9 patches, and predicting the locations of 8 patches with respect to the central patch of the image. [7]

By following the principles of self-supervision and prior work for 2D datasets, we design a pretext task to learn 3D representations and train a convolutional neural network to solve the pretext task. The model is then repurposed to solve object classification tasks.

1.3 Approach

In this project we design a self-supervised learning pretext task method for neural networks operating on raw point cloud data in which a neural network is trained to predict the rotation angles of a 3D model. The proposed method can be used for any 3D dataset, can be applied to any 3D network architecture and can be flexibly used to pre-train any deep learning model operating on raw point clouds for other downstream tasks. Through multiple experiments and qualitative analysis, we show that our pretext task learns 3D representations for the ModelNet10 and ModelNet40 datasets. [8] The designed pretext task

shows improved performance and data efficiency in a downstream object classification task. We also test our model for the model classification task in a transfer learning setting with the ShapeNet dataset.[9]

The main contributions of our project are:

- Design of a pretext task that learns 3D representations and uses a lower percentage of the training set.
- Generalization of prior work with a pretext task that can be applied to any point cloud dataset and that is architecture agnostic.
- Suite of experiments that set the groundwork for evaluating the performance of 3D pretext tasks on downstream tasks using transfer learning settings.

2. Methodology

2.1 Neural Net Architecture

Point clouds, by definition, have an irregular format, which makes it a challenge to apply any sort of structure-dependent neural architectures, like convolutional neural nets. As a result, point clouds are often transformed into regular 3D voxel grids or collections of images before performing a learning task. These transformations, however, are known to be computationally expensive and don't provide the best results. In order to account for this, we decided to use the PointNet architecture (Figure 1), which can be applied directly on point clouds.

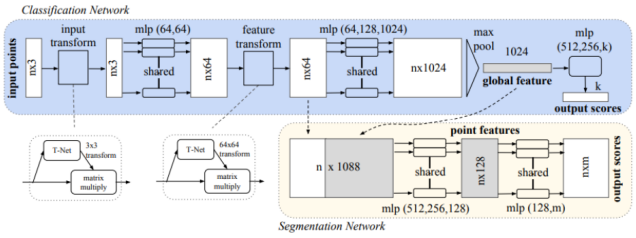

Fig. 1. Architecture for PointNet model.

The first transformation network shown in Figure 1 is a mini PointNet which takes a point cloud with n points and 3 dimensions (x, y, and z) (n x 3 matrix) and predicts an input-dependent 3 × 3 matrix transformation matrix. This is a type of regression network that is composed of a three-layered MLP with 64, 128, and 1024 neurons that is shared amongst all the points. This is followed by a max pooling operation across all the points and two fully connected layers of sizes 512 and 256. All layers, except the last one, use a ReLU activation function. The output from this transformation network is then multiplied with the n x 3 input. The new n x 3 matrix is then passed through a shared MLP (64, 64) which gives an n x 64 output. This output is passed through the second transformation network, which has the same architecture as the first transformation network, except that it predicts and multiplies a 64 x 64 matrix. The output from this transformation network is then passed through another shared MLP (64, 128, 1024) which results in an n x 1024 matrix. This is followed by a max pooling operation across all the n points to give a global feature vector of length 1024. This vector is used as the learned "embedding" for the classification task. It is important to note that a shared MLP means that the mapping weights were independent and identical for each of the n points. While training the network, a dropout with keep ratio 0.7 was used on the last fully connected layer. The adam optimizer was used with an initial learning rate of 0.005 and batch size of 32. This learning rate was divided by 2 every 20 epochs.

2.2 Datasets

Our model's performance was evaluated on the downstream classification tasks. For the downstream classification task, we used the ModelNet10 and ModelNet40 datasets, which include synthetic CAD-generated point clouds belonging to 10 and 40 different classes respectively (Figure 2). [8]

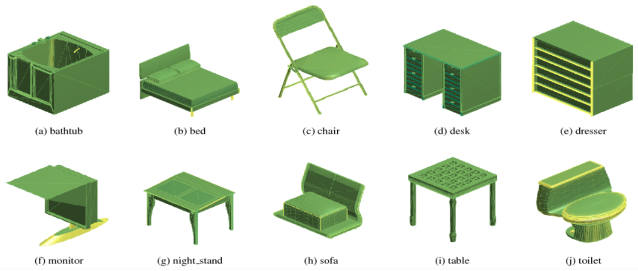

Fig. 2. Class labels in the ModelNet10 dataset.

Each category contains 150-800 models across the training and testing datasets. The ModelNet10 dataset consists of 4,899 unique 3D models across 10 categories. The ModelNet10 and ModelNet40 datasets were chosen because of their widespread use amongst researchers in this field, making it easy to benchmark the performance of our pretext task. In addition, it is extremely easy to find rotational transformations as well as the predetermined training and testing splits of the ModelNet datasets online.

For the transfer learning scenario, we used the ShapeNet dataset, which includes 16,000 3D models belonging to 16 classes. [9]

2.3 Pretext Task

In order to learn an "embedding" or "metric" that would be fundamental to 3D Point Clouds, we designed a rotational pretext task. The pretext task involved two sets of rotation for each point cloud in the training set. In Step 1, the angles between 0 and 90 degrees were divided into 6 uniform buckets. The model was then rotated about the x-axis by a randomly chosen rotation angle within a bucket (Figure 3). This step was repeated for each of the 6 buckets, generating 6 different rotated models. For Step 2, 10 equally-spaced angles were generated between 0 and 90 degrees. Then, each of the 6 models from the first step were then rotated about the y-axis by these 10 equally-spaced angles (Figure 4). This resulted in 10 different models for each of the 6 rotational buckets in the x-axis, giving a total of 60 different models for each point cloud in the training set.

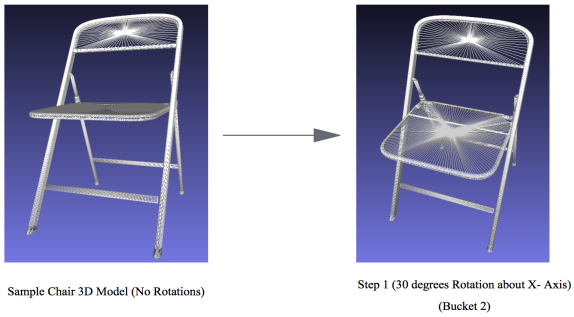

Fig. 3. Step 1 of the Pretext Task dataset generation.

The goal of the designed pretext task is to extract or learn information about each class in the training set that will aid in distinguishing these classes from one another. In order to do that, the information learned by the network should be able to represent a class meaningfully. If the pretext task is too easy, then the learned representations will be trivial and won't provide any meaningful information. In order to avoid trivial solutions, random rotational noise was added to each of the 60 point clouds generated after the two rotational steps. The added noise involved rotating each individual ing each individual point by a random value between 0 and 4.5 degrees (within 5 % of 90 degrees).

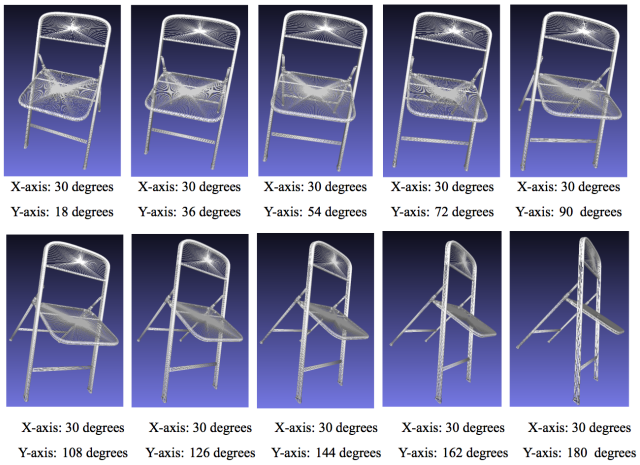

Fig. 4. Step 2 of the Pretext Task dataset generation.

The rotated point clouds were used as inputs to the PointNet architecture and the output was the rotational bucket. The network was thus trained to classify the approximate degree of rotation about the x-axis. After the network finished training, the weights were saved and then used as the initialization for the classification task. The classification network was trained on only 10% of the ModelNet10 dataset.

A similar process was repeated with the ModelNet40 dataset, in which the network was trained on the pretext task dataset for ModelNet40 and these weights were then used as the initialization for the classification task. For ModelNet40, the classification network was trained on 5%, 10%, 25%, 50%, and 10% of the ModelNet dataset.

3. Results

TABLE 1: Baseline vs Pretext Task Training

Task	Dataset	Model	Output	Accuracy
Baseline	ModelNet10	PointNet	10 Classes	94%
Pretext Task	ModelNet10	PointNet	Rotation Bucket (without Noise)	92%
Pretext Task	ModelNet10	PointNet	Rotation Bucket (with Noise)	88%

Table 1 shows the training accuracy of Baseline and Pretext Task. Since the accuracy of Pretext task with the added noise is close to the Baseline task, we can conclude that the pretext task is adaptive in this situation.

Then, we generated learned embeddings using our pretext task which were able to differentiate between the 10 classes. The embedding, in our case, is a l024-dimensional global feature vector. By using random initialization as our benchmark, we can find out that our pretext task is able to learn the differences between the different classes. As shown in Figure 5, similar classes converge to a relatively smaller area after implementation of the pretext task.

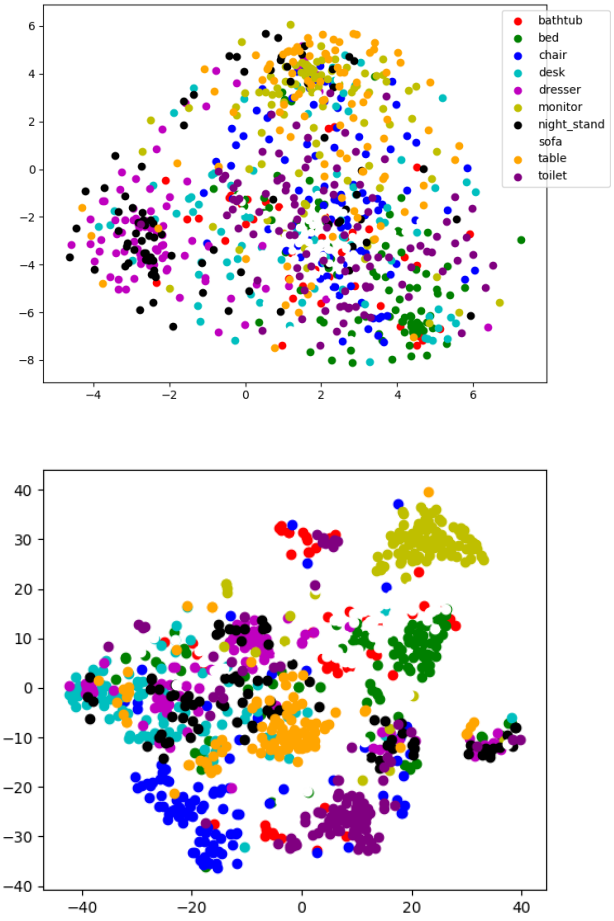

Fig. 5: (Top) t-SNE plot of embeddings generated by random weights. (Bottom) t-SNE plot of embeddings generated by pretext task.

Using the weights from the network trained on the pretext task as the initialization parameter for the classification task, we obtained a higher accuracy for the classification task that was trained on only 10% of the dataset. The accuracy on the ModelNet10 dataset using the pretrained weights was 81% while the accuracy using the randomly initialized weights was 78% (Figure 6).

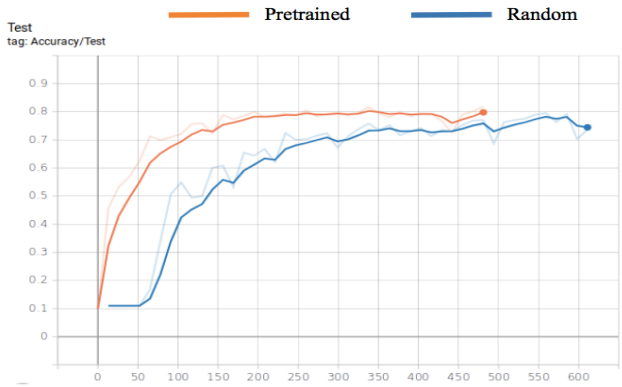

Fig. 6. Test accuracy on 10% of ModelNet10 data.

4. Discussion and Detailed Analysis

Over the past few years, with the widespread availability of cheap 3D scanners, point clouds have become one of the simplest and most cost-effective ways to represent 3D spatial data. Storing information as a collection of spatial coordinates can help in saving space, since many objects do not fill up a lot of the environment. At its core, a point cloud is just an unordered collection of objects in 3 dimensions (x, y and z) that describe some structure in space. This unordered set also has a distance metric defined on it, which makes point clouds a metric space. The PointNet architecture uses this metric to define operations that can be applied directly onto point clouds. For classification task, point clouds have 2 unique properties, which make it difficult to directly apply architectures developed for 2D images. These are:

1. Order Invariance: A point cloud made up of N points has N! permutations. Any neural net architecture should, therefore, also be invariant to the N! permutations.
2. Transformation Invariance: Classification outputs should be unchanged if the point cloud undergoes transformations like rotation and translation.

In order to be invariant to input permutations, PointNet uses a max pooling operation to generate the

global feature vector. This operation is symmetric in nature, i.e., given *n* arguments, the output of the operation is the same regardless of the order of the *n* arguments. To provide transformation invariance, the PointNet architecture uses two different features transformations that provide pose normalization. This is motivated by the success of Spatial Transformer Networks (STNs) for pose normalization in 2D images. The second transformation outputs a 64 x 64 matrix and because of the increased number of trainable parameters, the probability of overfitting also increases. In order to counter this, a regularization term is added to the loss function that drives the resulting 64-by-64 transformation matrix to be close to an orthogonal transformation. Thus, the PointNet architecture is well-suited to our goals of designing a pretext task on 3D point clouds.

In order to develop a suitable pretext task, the order and transformation invariance of point clouds were taken into consideration. A piercing question related to the rotational pretext task that was considered was that since the PointNet architecture is invariant to rotational transformations, will the learned "embedding" be trivial and not provide meaningful information about the different classes? A two-step rotational task was used as a solution. The rotations about the y-axis served as a way to augment the training data set as well as increase the complexity of the task. Moreover, the output of the PointNet training was a bucket of rotation angles as opposed to the classes of the point clouds. This ensured that the rotational invariance of the architecture didn't affect the information learned from the pretext task. As a third measure against the rotational invariance, random rotational noise was added to each point cloud after the two-step rotation process. The lower accuracy (88%) obtained for the rotational pretext task (with noise) as compared to the baseline classification (94%) proves that the designed task was not trivial.

It is clear from Figure 5 that the learned embeddings for most of the 10 classes are well differentiated. The embeddings for "chair" and "toilet" are very close to one another, which makes intuitive sense since chairs and toilets have very similar forms. The two classes which were not very well differentiated are "bathtub" and "night_stand". This results from the network not being able to distinguish or represent the local structures for these two classes. In order to understand this better, the embeddings generated from the baseline classification task were visualized using t-SNE plots (Figure 7). A comparison between the two plots revealed that the "night_stand" class was not distinguishable in both the tasks, which might be a result of the shortcomings of the PointNet architecture rather than the pretext task. However, as shown in Figure 7, the "bathtub" class was very easily distinguishable in the baseline classification task.

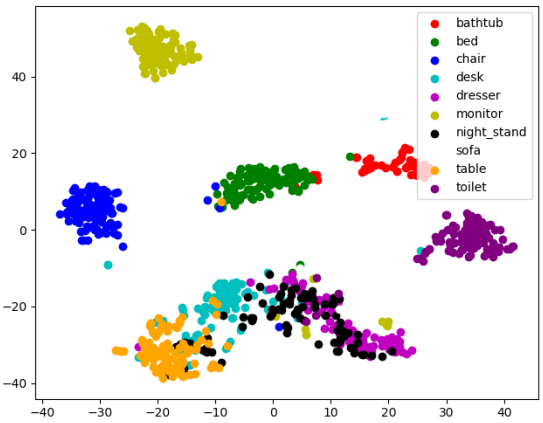

Fig.7. t-SNE plot of embeddings generated by baseline classification

This proves that the rotational pretext task needs improvements to be able to successfully distinguish multiple classes. However, as shown in the results in Figure 6, the weights generated from the pretext tasks served as a better initialization parameter. The classification accuracy obtained was higher for the same number of epochs. Since the training set for this consisted of only 10% of the total dataset, we can conclude that our pretext task performs better even with limited data. This can be extremely useful in studying rare diseases or other similar cases where a large amount of data is not available.

To further our analysis, the designed pretext task was also evaluated on the ModelNet40 dataset to ensure scalability with a large number of classes . The classification model was trained with 5%, 10%, 25%, 50%, and 100% of the ModelNet40 dataset, to show its performance with a smaller dataset. As the data efficiency curve in Figure 7 shows, the pretrained weights perform better than a random initialization in most scenarios. The objective of this curve is to show that with the pretrained weights, the model requires less labelled training data to achieve a high accuracy.

For example, with 10% of the labelled training data the model with pretrained weights achieves 76.74% accuracy, while the supervised baseline model with random weights achieves a similar accuracy of 76.69% with 25% of the labelled training data. This will have

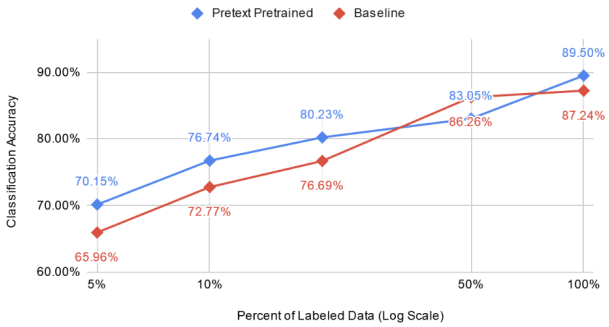

Fig. 8. Data efficiency curve for ModelNet40 data.

many applications in areas where annotation of 3D data is time-consuming and expensive, like development self-driving algorithms.

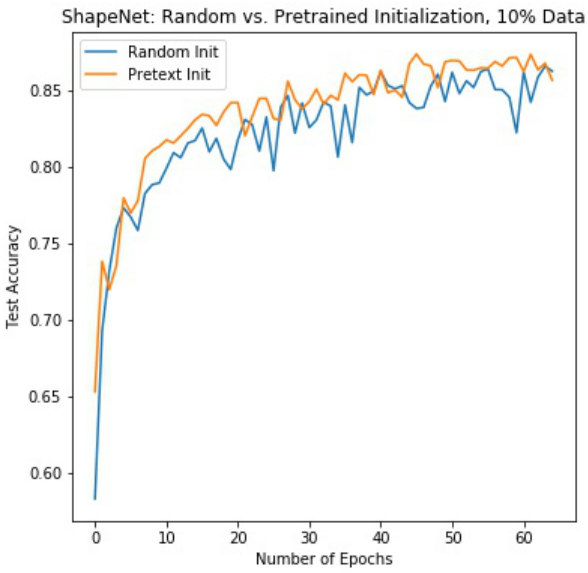

Fig. 9 Test accuracy on ShapeNet dataset.

In order to show that the learned representations carry useful information about the classes, we evaluated the performance of the task in a transfer learning scenario. This involved using the weights generated from the trained pretext task using the ModelNet40 dataset to initialize a classification on a completely different dataset (ShapeNet) [9]. As shown in Fig. 9, the pretext task performed slightly better than the supervised learning task with random initialization while using only 10% of the dataset as the training set. This proves that embeddings and weights learned from the ModelNet40 dataset can be applied to ShapeNet and possibly many other datasets.

In this paper, we proposed a self-supervised method for learning representations from raw point cloud data. We also demonstrated the effectiveness of the learned representations in classification tasks as well as in transfer learning scenarios. The methods described in this paper can be used to improve accuracy when working with limited or unlabelled datasets. In future studies, we would like to evaluate our pretext task with different underlying architectures to produce better results. In addition, we would also like to compare other potential 3D pretext tasks.

References

[1] Wikipedia.org. (2019). Machine vision. [online] Available at: https://en.wikipedia.org/wiki/Machine_vision

[2] Ahmed, E., Saint, A., Shabayek, A.E.R., Cherenkova, K., Das, R., Gusev, G., Aouada, D. and Ottersten, B., 2018. A survey on Deep Learning Advances on Different 3D Data Representations. arXiv preprint arXiv:1808.01462.

[3] Qi, C.R., Su, H., Mo, K. and Guibas, L.J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 652-660).

[4] The Robot Report Staff (2019). Scale AI raises $100M Series C for annotation services to train AI. The Robot Report. Available at: https://www.therobotreport.com/scale-ai-series-c-annotation-services-ai-robotics/

[5] Jing, L. and Tian, Y., 2019. Self-supervised visual feature learning with deep neural networks: A survey. arXiv preprint arXiv:1902.06162.

[6] Zhang, R., Isola, P. and Efros, A.A., 2016, October. Colorful image colorization. In European conference on computer vision (pp. 649-666). Springer, Cham.

[7] Noroozi, M. and Favaro, P., 2016, October. Unsupervised learning of visual representations by solving jigsaw puzzles. In European Conference on Computer Vision (pp. 69-84). Springer, Cham.

[8] Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H. and Xiao, J., 2015. Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012.

[9] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X. and Xiao, J., 2015. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1912-1920).