Self-Supervised Learning of Local Features in 3D Point Clouds

Ali Thabet* Humam Alwassel* Bernard Ghanem
King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia
{ali.thabet, humam.alwassel, bernard.ghanem}@kaust.edu.sa

Abstract

We present a self-supervised task on point clouds, in order to learn meaningful point-wise features that encode local structure around each point. Our self-supervised network, operates directly on unstructured/unordered point clouds. Using a multi-layer RNN, our architecture predicts the next point in a point sequence created by a popular and fast Space Filling Curve, the Morton-order curve. The final RNN state (coined Morton feature) is versatile and can be used in generic 3D tasks on point clouds. Our experiments show how our self-supervised task results in features that are useful for 3D segmentation tasks, and generalize well between datasets. We show how Morton features can be used to significantly improve performance ($+3\%$ *for* 2 *popular algorithms) in semantic segmentation of point clouds on the challenging and large-scale S3DIS dataset. We also show how our self-supervised network pretrained on S3DIS transfers well to another large-scale dataset, vKITTI, leading to* 11% *improvement. Our code is publicly available.*[1]

1. Introduction

Given their massive success with 2D data, deep learning algorithms are a natural option to solve 3D computer vision problems. The most common form of 3D data comes in the form of unstructured 3D point clouds. To that extent, PointNet [9] paved the way to applying deep learning directly to point clouds and served as a gateway for a variety of works in this area. These works generally fall into 2 categories: point-wise networks and point convolutions. In the former, point features are extracted usually through fully-connected or 1D convolutions layers, and later aggregated using simple global feature pooling or more complex RNN architectures. Point convolution methods define convolution kernels that operate on point neighborhoods directly.

We propose learning a new type of point-wise feature by using points and a structured neighborhood around them.

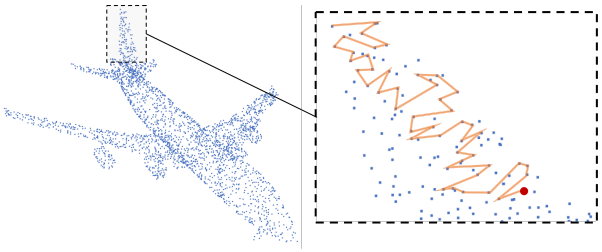

Figure 1. **Self-Supervision of 3D Point Clouds.** We propose to learn a point-wise feature representation by leveraging space filling curves, namely the Morton- or Z-order. Given this ordering, we setup a self-supervised task that aims to predict a point (the red point) from the sequence points (the orange points). Features learned from this self-supervision can be used to improve performance in point cloud semantic segmentation.

We define this neighborhood based on a predefined meaningful geometric ordering for 3D points, namely the well-known Morton-order (also known as Z-order) Space Filling Curve (SFC). Given this ordering, a self-supervised task of point prediction can be exploited to extract local point-wise features, which can be later used in the common 3D tasks of semantic segmentation (refer to Figure 1).

Contributions: We summarize our contributions as threefold. **(1)** We propose a self-supervised task and accompanying network to learn point-wise features in 3D point clouds; we denote these learned features as Morton features. **(2)** Since Morton features are learned in a self-supervised manner, we show how they can be easily incorporated in popular 3D tasks. Specifically, we show how they can be used in semantic segmentation pipelines to substantially increase their performance by $+3\%$ on the challenging S3DIS [1] dataset. **(3)** We show how Morton features trained on S3DIS successfully transfer to a different dataset, vKITTI [4], where we improve by 11% in semantic segmentation.

2. Related Work

Deep Learning for Point Clouds. We summarize approaches that operate directly on unordered point clouds. PointNet [9] pioneered the way for deep learning methods to be applied to unordered point clouds. The authors

*equal contribution
[1]https://github.com/alitabet/morton-net

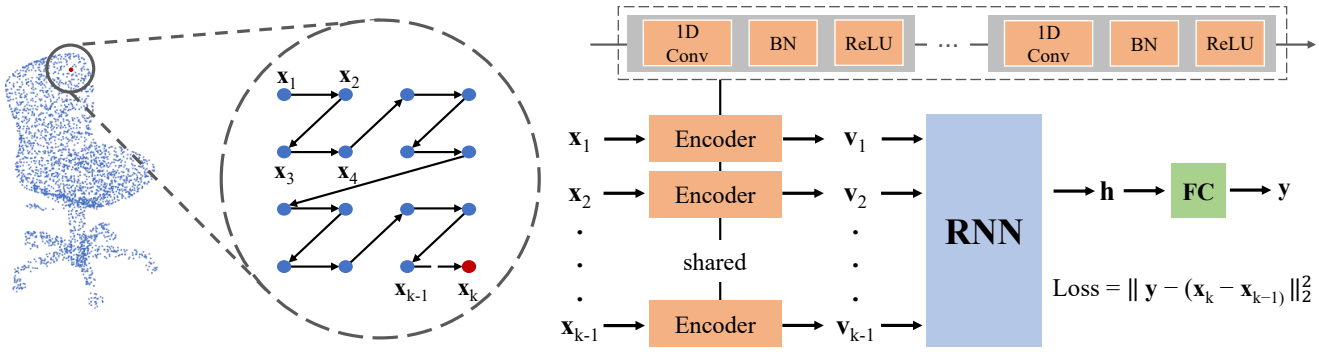

Figure 2. **Learning Morton Features.** Our architecture aims to learn local features of unstructured 3D point clouds. Our model is trained in a self-supervised fashion to predict the displacement to the next point in a Z-ordered sequence. Given an unstructured point cloud, we first generate multiple Z-ordered sequences that end at each point (*e.g.* the red point $\mathbf{x}_k$). We encode each point $\mathbf{x}_i$ in the sequence (except $\mathbf{x}_k$) into a higher dimensional vector $\mathbf{v}_i$ with a series of 1D convolutions, batch normalization, and ReLU. The sequence of vectors $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{k-1})$ are then fed into a multi-layer RNN, whose final state $\mathbf{h}$ is transformed by a fully connected layer to produce the final displacement prediction $\mathbf{y}$. We regress $\mathbf{y}$ to the ground truth displacement $(\mathbf{x}_k - \mathbf{x}_{k-1})$ for each Z-ordered sequence.

show the benefit of using this data modality as opposed to either voxelized or multi-view methods. PointNet takes point cloud chunks, computes point features, and aggregates those features with an order-invariant operation like max pooling. Pooled features are used to either label the whole chunk (classification) or each point (segmentation). Inspired by this line of work, new research presents better ways to compute and aggregate point features by either looking at more local context [10, 4] or using more complex RNN based methods [6, 16]. Recent approaches try to leverage better point neighborhoods [17, 7], or construct complex point based kernels [12].

Self-Supervision. Self-supervision is a form of unsupervised learning, where supervision originates from the data itself. The principal idea is to set a task that only depends on the data available. This task serves as a proxy to learn semantic representations. With images, self-supervision has been applied to learn relative patch locations from the same image [2], learn colorization [18], design examplar CNNs [3], and learn image rotations [5]. In video, examples of self-supervision work are learning the next frame in a video sequence [14] and tracking by colorization [15]. For 3D data, few works use unsupervised methods to learn volumetric representations of 3D data [13, 11]. In this paper, we present a new method to compute meaningful point features for unstructured point clouds. We compute these features through a self-supervised task that does not require semantic labels. Our features can represent the local structure around each point. We show how these features can be incorporated to perform point cloud semantic segmentation, and how they are generalizable between datasets.

3. Methodology

In this section, we formulate a self-supervised proxy task to learn point features (Figure 2). We highlight the process to generate the self-supervised training data from any point cloud, and apply it to the large-scale and dense S3DIS dataset [1]. Finally, we detail our proposed model architecture to solve the self-supervised task.

3.1. Predict-the-Next-Point Self-Supervised Task

Given a k-long Z-ordered sequence $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k)$, the task is to predict $\mathbf{x}_k$ from the previous $(k-1)$ points in the sequence. To stabilize the learning process, we consider the equivalent task where we predict the displacement $\mathbf{x}_k - \mathbf{x}_{k-1}$ given the $(k-1)$-long subsequence $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{k-1})$. The Z-order gives us a stable structure to learn from any unstructured point cloud. We turn to self supervision to learn this task since it does not require annotated data. Next, we detail how Z-ordered sequences can be generated from an unstructured point cloud.

3.2. Z-ordered Sequence Generation

Let $\mathcal{S}_r(\mathbf{x})$ be the set of points that lie in a neighborhood of radius r around the point $\mathbf{x}$. We refer to $\mathcal{S}_r(\mathbf{x})$ as the support of $\mathbf{x}$. Given a point $\mathbf{x}$ in our 3D point cloud, we generate Z-ordered sequences that end at $\mathbf{x}$ in the following way: we order all points in $\mathcal{S}_r(\mathbf{x})$ according to their Z-order [8], then we randomly sample $(k-1)$ points from $\mathcal{S}_r(\mathbf{x})$ that have a smaller Z-order compared to that of $\mathbf{x}$. Finally, we construct a Z-ordered sequence of length k that ends at $\mathbf{x}$, *i.e.* the $(k-1)$ sampled points plus $\mathbf{x}$. We generate m different Z-ordered sequences in the same manner for every point $\mathbf{x}$ in the point cloud. Our sequence generation procedure guarantees that we cover each point with multiple sequences, allowing us to capture the different aspects of local structure in the point cloud.

3.3. Proposed Architecture

The input to our model is a $(k-1)$-long Z-ordered sequence of 3D points, $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{k-1})$, and the expected

Rand.	Coordinate-wise			**Ours**		
	x	y	z	$k = 20$	$k = 60$	$k = 100$
36.6	37.6	36.9	37.9	34.0	46.6	**84.0**

Table 1. **Ours vs. Baselines**. Test accuracy comparison between our architecture and four baselines (random ordering and three coordinate-wise orderings) on the predict-the-next-point task. We show the results for different sequence lengths k. All baselines use $k = 100$. Ours with $k = 100$ gives the best results.

output is the displacement to the next point, *i.e.* $(\mathbf{x}_k - \mathbf{x}_{k-1})$. We encode each 3D point $\mathbf{x}_i$ into a higher dimensional vector $\mathbf{v}_i$ using a series of spatial encoding layers, each of which consists of a set of 1D convolutions followed by batch normalization and a ReLU activation function. Then, we feed the high dimensional vector sequence ($\mathbf{v}_1$, $\mathbf{v}_2$, ..., $\mathbf{v}_{k-1}$) to a multi-layer RNN. Finally, we transform the last RNN hidden state using a fully connected layer to produce a 3D final output $\mathbf{y}$, *i.e.* an estimate of the spatial displacement needed to reach the next point in the sequence. Figure 2 gives an overview of our proposed architecture.

3.4. Implementation Details

We select the minimum radius r for which we have at least $2 \times k$ points in the support set $\mathcal{S}_r(\mathbf{x})$. We experiment with different sequence length values, namely $k = 20$, 60, and 100. We empirically set the number of sequences per point to $m = 5$. For our architecture, we use 4 encoding layers with 64 1D-convolution kernels in each. We choose a GRU with 4 layers and hidden state size of 200, and use the hidden state as our Morton features. We use an Adam optimizer, train for 40 epochs, and set the initial learning rate to 10^{-3}. We decrease the learning rate with a decay factor of 0.9 when the validation loss does not improve for 2 epochs. We pick the model with the best validation loss.

4. Experiments

We train our model on the Predict-the-Next-Point task (Section 3.1) on the S3DIS dataset [1]. We choose S3DIS, since it is a large-scale dataset with high density point clouds and a large number of object instances. These characteristics translate into a large variety of Z-ordered sequences for our model. We incorporate the learned Morton features into point cloud semantic segmentation. We show how Morton features trained on S3DIS enhance semantic segmentation on the same dataset and transfer well to vKITTI [4], a dataset of outdoor scenes.

4.1. Dataset Details

S3DIS [1]. The Stanford Large-Scale 3D Indoor Spaces (S3DIS) dataset consists of indoor 3D point clouds of 6 large areas. The indoor scans cover an area of 6020 meters and contain over 215 million points. The data is semantically separated into 272 rooms and annotated with 12 semantic elements and an additional label for clutter. S3DIS is typically used for indoor semantic segmentation.

vKITTI [4]. Virtual-KITTI (vKITTI) is a synthetic large outdoor dataset with 13 semantic classes from urban scenes. vKITTI imitates data from the real-world KITTI dataset. It contains data from 50 high resolution scenes. This dataset is used for a variety of tasks, where the most common one for point clouds is semantic segmentation.

4.2. Learning Morton Features

Experimental Settings. We study the performance of our model on the self-supervised task on S3DIS. We discard the semantic label information in S3DIS and only use the unlabelled point clouds. We train our model on a subset of the original data. In particular, we select 8 rooms from the training subset and 2 from the testing subset. This constitutes less than 4% of all rooms. We first generate the k-long Z-ordered sequences for all these rooms, and later subsample 40% of the training sequences for computational efficiency.

Baseline Methods. To demonstrate the effectiveness of learning from Morton-ordered sequences, we train baseline models on the same predicting-the-next-point task from sequences ordered using different mechanisms. The *Random Sequence Baseline* is trained on sequences of points that are randomly ordered. We choose this model for reference, so as to show that learning this self-supervised task is more meaningful when the sequence is ordered. The *Coordinate-Ordered Sequence Baselines* train three models on point sequences ordered according to their x-, y-, and z-coordinates. We choose these models to show that Morton-order is a better ordering choice.

Evaluation Metric. To measure the accuracy of our model and the baselines in predicting-the-next-point, we consider a prediction to be correct if it is within a ball of radius ρ from the correct next point in the sequence. In other words, if the ground truth displacement is $(\mathbf{x}_k - \mathbf{x}_{k-1})$, and the the model predicts a displacement value $\mathbf{y}$, then the prediction is correct if and only if $\|\mathbf{y} - (\mathbf{x}_k - \mathbf{x}_{k-1})\|_2 \leq \rho$. We set $\rho = 0.02$ in our experiments and report the performance as the average accuracy on the testing sequences.

Results. Table 1 reports the test accuracy of our model against the baselines on S3DIS. Our model is consistently superior to both baselines. This shows the importance of the Z-order in learning local point features from unstructured point clouds. Additionally, we show the effect of the parameter k (the sequence length) on the accuracy. As k increases, our model becomes more accurate in predicting the next point. We attribute this correlation between k and accuracy to the fact that with longer Z-ordered sequences, our model has access to a bigger support set $\mathcal{S}_r(\mathbf{x})$ and thus can encode more complex local structures around each point. However, while increasing k beyond 100 would result in a more accurate model, the increase in performance comes at

Method	mIoU	mAcc	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	book.	board	clutter
PN [9]	41.1	49.0	88.8	97.3	69.8	0.05	3.92	**49.3**	10.8	58.9	52.6	**5.8**	40.3	26.3	32.2
[9] + **Morton features**	**44.4**	**63.1**	**90.3**	**97.9**	**74.6**	**0.11**	**9.55**	47.4	**17.7**	**64.2**	**57.9**	1.42	**43.2**	**33.5**	**40.1**
RSNet [6]	51.9	59.4	**93.3**	98.4	79.2	0.0	15.8	45.4	**50.1**	67.9	65.5	**52.5**	22.5	**41.0**	43.6
[6] + **Morton features**	**55.0**	**61.2**	92.5	**98.5**	**81.4**	0.0	**24.2**	**47.6**	50.0	**70.8**	**76.7**	31.6	**58.2**	38.5	**45.1**

Table 2. **Semantic Segmentation on S3DIS**. Results of applying Morton features to the S3DIS semantic segmentation task for test Area 5. The first sub-table shows results using PointNet and PointNet with Morton features. The second sub-table compares RSNet with and without Morton features. Performance is measured by mean intersection of union (mIoU) and mean accuracy (mAcc) across classes. We see clear improvement in mIoU when adding Morton features to both PointNet and RSNet. Bold represent best numbers on each sub-table.

the cost of making the model computationally expensive, since RNNs process data sequentially. Thus, we choose $k = 100$ for all remaining experiments.

4.3. Morton Features for 3D Tasks

We now use our trained model in 2 different experiments: semantic segmentation on S3DIS and semantic segmentation on vKITTI. In each experiment we choose a point cloud method and enhance it with Morton features. We show this enhancement provides significant boost in the method's performance. Our aim is to demonstrate the power of the Morton features we learned by showing that: (1) they are effective for 3D point-based semantic segmentation and (2) they generalize well across different datasets.

Experimental Settings. We use our model (with $k = 100$) trained on S3DIS to extract features for semantic segmentation in S3DIS and vKITTI. Using the procedure detailed in Section 3.2 to extract features, we generate Z-ordered sequences for the two datasets, pass these sequences to our model, and aggregate the hidden state of the RNN. Here, we use the same model pre-trained on the unlabelled point clouds of S3DIS and do not finetune on other datasets.

Evaluation Metrics. We measure performance in all experiments using the mean point intersection over union (mIoU) across classes (the preferred metric for semantic segmentation) and the mean accuracy (mAcc).

Models. For the choice of segmentation algorithms, we select the classical PointNet [9] and RSNet [6], all with readily available code. For PointNet, we concatenate our precomputed Morton features to those extracted in PointNet before the point classifier. In RSNet, we concatenate our pre-computed Morton features with their point features before they are fed to their RNN layers.

Results on S3DIS. Table 2 presents our results for semantic segmentation on S3DIS. We use the same data processing as in [6], and evaluate results on Area 5. We note how augmenting both PointNet and RSNet with Morton features boosts their performance by over 3 points in mIoU. These results show training our model with S3DIS translates to meaningful information about local point structure, which can be used to significantly improve semantic segmentation performance.We notice significant boost in classes like bookcase, table, and chair, where we believe the local information from Morton features provides information otherwise inaccessible to the networks.

Method	OA	mIoU	mAcc
PN [9]	79.7	34.4	47.0
PN + **Morton features**	**91.8**	**45.4**	**63.5**

Table 3. **Semantic Segmentation on vKITTI**. Results of applying Morton features to the vKITTI semantic segmentation task. Adding Morton features to PointNet significantly increases its performance. Bold numbers represent the best numbers.

Results on vKITTI. In this experiment, we use our model pre-trained on S3DIS to extract features on vKITTI point clouds, and use these features in a semantic segmentation pipeline. Experiments in vKITTI are done using the 6-fold cross-validation splits suggested by [4]. Table 3 shows empirical evidence for a critical strength of our model. By adding Morton features to PointNet (simple model), we improve its performance by 11 points in mIoU. We believe one of the reasons why semantic segmentation is challenging in vKITTI is the low sampling density of points in this dataset. Since Morton features are trained using a very dense dataset (S3DIS), they include local information that is otherwise missed during preprocessing in vKITTI. This information transfers well when computing features on a different, more sparse dataset. Although we do not perform specific ablation studies to prove this fact, we believe this additional information is responsible for the performance increase brought about by incorporating Morton features.

5. Conclusion

We presented a self-supervised approach to learning local point-wise features from 3D point clouds. We showed the importance of learning from Morton-ordered sequences, and demonstrated the effectiveness of Morton features for 3D tasks. We showed significant improvement when augmenting other methods with our features, and showed that our features can generalize well to a different dataset. Although we only used them for segmentation, we believe that Morton features are generic and effective in encoding local point structure to enhance other 3D tasks.

References

[1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1, 2, 3

[2] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015. 2

[3] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2016. 2

[4] Francis Engelmann, Theodora Kontogianni, Alexander Hermans, and Bastian Leibe. Exploring spatial context for 3d semantic segmentation of point clouds. feb 2018. 1, 2, 3, 4

[5] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 2

[6] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2626–2635, 2018. 2, 4

[7] Guohao Li, Matthias Müller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 2

[8] Guy M Morton. A computer oriented geodetic data base and a new technique in file sequencing. 1966. 2

[9] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4, 2017. 1, 4

[10] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017. 2

[11] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096, 2017. 2

[12] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. *ArXiv*, abs/1904.08889, 2019. 2

[13] Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2635–2643, 2017. 2

[14] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating the future by watching unlabeled video. *arXiv preprint arXiv:1504.08023*, 2, 2015. 2

[15] Carl Vondrick, Abhinav Shrivastava, Alireza Fathi, Sergio Guadarrama, and Kevin Murphy. Tracking emerges by colorizing videos. *arXiv preprint arXiv:1806.09594*, 2018. 2

[16] Xiaoqing Ye, Jiamao Li, Hexiao Huang, Liang Du, and Xiaolin Zhang. 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In *European Conference on Computer Vision*, pages 415–430. Springer, 2018. 2

[17] Ziwei Liu Sanjay E. Sarma Michael M. Bronstein Justin M. Solomon Yue Wang, Yongbin Sun. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 2

[18] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016. 2