Point2Vec for Self-Supervised Representation Learning on Point Clouds

Karim Abou Zeid*, Jonas Schult*, Alexander Hermans, and Bastian Leibe

RWTH Aachen University, Germany
{abouzeid,schult,hermans,leibe}@vision.rwth-aachen.de
https://vision.rwth-aachen.de/point2vec

Abstract. Recently, the self-supervised learning framework data2vec has shown inspiring performance for various modalities using a masked student–teacher approach. However, it remains open whether such a framework generalizes to the unique challenges of 3D point clouds. To answer this question, we extend data2vec to the point cloud domain and report encouraging results on several downstream tasks. In an in-depth analysis, we discover that the leakage of positional information reveals the overall object shape to the student even under heavy masking and thus hampers data2vec to learn strong representations for point clouds. We address this 3D-specific shortcoming by proposing point2vec, which unleashes the full potential of data2vec-like pre-training on point clouds. Our experiments show that point2vec outperforms other self-supervised methods on shape classification and few-shot learning on ModelNet40 and ScanObjectNN, while achieving competitive results on part segmentation on ShapeNetParts. These results suggest that the learned representations are strong, highlighting point2vec as a promising direction for self-supervised learning of point cloud representations.

1 Introduction

In this work, we address the task of self-supervised representation learning on 3D point clouds. With the ever increasing availability of affordable consumer-grade 3D sensors, point clouds are becoming a widely adopted data representation for capturing real-world objects and environments [1, 6–8, 15]. They provide accurate 3D geometry information, making them a valuable input for many applications in the field of robotics, autonomous driving [7, 8], and AR/VR applications. The 3D computer vision community has made impressive progress by developing 3D-centric approaches which directly process 3D point clouds to semantically understand 3D objects and environments [14, 35, 36, 39]. However, these approaches typically rely on fully-supervised training *from scratch* [46], requiring time-consuming and labor-intensive human annotations. For example, semantically annotating a single room-scale scene of the ScanNet dataset takes about 22 minutes [15]. This results in a lack of large-scale annotated point cloud datasets, making it challenging to learn strong representations from limited data.

[*] Equal contribution.

At the same time, self-supervised training has shown impressive results in natural language processing [16, 47], speech [3, 26], and 2D vision [2, 9, 12, 22, 23], enabling learning of meaningful representations from massive unlabeled datasets without any human annotations. Only recently, we have seen self-supervised methods being successfully applied to Transformer architectures for 2D vision [2, 9, 23] and 3D point clouds [34, 49, 51]. Baevski *et al.* propose data2vec [2], a modality-agnostic self-supervised learning framework showing competitive performance in speech recognition, image classification, and natural language understanding. Data2vec uses a joint-embedding architecture [2, 9, 22] with a *student* Transformer encoder and a *teacher* network parameterized as the exponential moving average of the student weights. Specifically, the teacher first predicts latent representations using an uncorrupted view of the input, which the student network then predicts from a masked view of the same input.

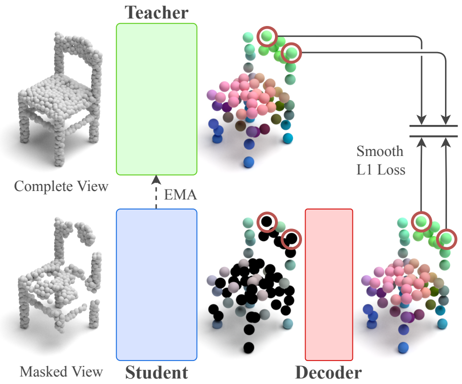


Fig. 1: Overview of point2vec. During training, a teacher network 🟩 predicts latent representations using a complete view of the point cloud. The student network 🟦 predicts the same representations, but from a partial view. A shallow decoder 🟥 then reconstructs the latent representations of masked regions ●, which we can use to train the student and the decoder, whereas the teacher uses an exponential moving average of the student weights.

In this paper, our aim is to apply data2vec-like pre-training to point clouds. The key difference to top-performing approaches for point cloud representation learning such as Point-MAE [34], Point-M2AE [51] and Point-BERT [49] is the target representation. The self-attention in the student Transformer encoder of data2vec generates *contextualized* feature targets that contain *global* information of the entire input. In contrast, Point-MAE [34] and Point-M2AE [51] explicitly reconstruct only *local* point cloud patches, and Point-BERT [49] is restricted to a fixed-sized vocabulary of token representations. To apply data2vec [2] on point clouds, we use the same underlying 3D-specific Transformer model as Point-BERT [49] and Point-MAE [34]. In experiments, we show that these modality-specific adaptations to data2vec already enable competitive performance compared to highly 3D-specific self-supervised approaches [29, 34, 44, 49, 51]. Encouraged by these promising results, we perform a subsequent analysis that reveals a crucial and point cloud specific shortcoming that restricts data2vec's representation learning capabilities: data2vec uses masked embeddings in the student network which carry positional information. Unlike images, text, and speech, the positional information in point clouds contains semantic meaning, namely 3D point locations (Fig. 4). Feeding masked embeddings with positional information into the student network therefore reveals the overall object shape to the student which makes the masking operation far less effective, as also reported by Pang *et al.* [34] in the context of masked autoencoders for point clouds. Based on this analysis, we propose point2vec that effectively addresses the leakage of positional information to the student and thus unleashes the full potential of data2vec-like pre-training for point clouds. To this end, we exclude masked embeddings from the student network. This prevents the overall object shape from being revealed, while also decreasing the computational cost. Instead, we introduce a shallow decoder which processes masked embeddings together with the student's outputs and which is trained to regress the representations of the teacher (Fig. 1).

Evaluating the quality of the learned representations on downstream tasks is a crucial step for analyzing self-supervised methods. After pre-training on the ShapeNet dataset [10], our experiments demonstrate that point2vec outperforms other self-super-vised methods on both the ModelNet40 [45] and ScanObjectNN [43] shape classification benchmarks. Additionally, point2vec achieves state-of-the-art performance on few-shot classification on ModelNet40 and competitive results on Part Segmentation on ShapeNetPart [48]. These findings suggest that the learned representations are strong and transferable, indicating that point2vec is a promising approach for self-supervised point cloud representation learning.

To summarize, our contributions are: **(1)** We extend the seminal work data2vec [2] to the point cloud domain. **(2)** In our experiments, we discover a crucial shortcoming of data2vec that hampers its representation learning capabilities for point clouds: Masked embeddings leak positional information to the student, revealing the overall object shapes even under heavy masking. **(3)** We propose point2vec which unleashes the full potential of data2vec-like pre-training for self-supervised representation learning by addressing the aforementioned shortcomings. Point2vec learns strong and transferable features in a self-supervised manner, outperforming self-supervised approaches on several downstream tasks.

2 Related Work

Self-Supervised Learning. Recently, self-supervised learning gained much attention due to its promise to learn meaningful data representations without any human annotations. At the heart of self-supervised learning is the *pretext* task, offering a vast range of diverse options. One such line of work investigates contrastive learning objectives [12, 24, 33, 41, 42], *i.e.* maximizing feature similarity across multiple views of the same training sample while simultaneously minimizing the similarity to other training samples. Contrastive learning approaches typically rely on a careful choice of data augmentations, negative sample mining, or large batch sizes [22]. Addressing these limitations, student–teacher approaches [2, 5, 9, 13, 22] follow a *joint-embedding* architecture, *i.e.* two copies of the same network are trained to produce similar latent representations for two views of the identical input. Among them and most important to our work is data2vec [2], which relies on a teacher first generating targets by predicting latent representations using the complete view of the input and a student which predicts these targets using only a *masked* view of the same input. Inspired by data2vec's flexibility across a wide range of modalities, in this paper, we seek to unlock the full potential of data2vec-like pre-training for point clouds by specifically taking the unique characteristics of point clouds into account.

Self-Supervised Learning on Point Clouds. The success of self-supervised learning in 2D vision [2, 4, 5, 9, 12, 22, 23, 33], natural language processing [2, 16], and speech [2, 3] has inspired a number of recent works proposing self-supervised learning frameworks for point cloud understanding tasks. Among them, contrastive self-supervised frameworks are typically deployed for room-scale pre-training. The pioneering work of Xie *et al.* [46] contrasts corresponding 3D points from multiple partial views of a reconstructed static scene, showing impressive improvements when fine-tuned on several scene-level downstream tasks. Extending upon this, Hou *et al.* [25] propose to leverage both point-level correspondences and spatial contexts of 3D scenes. In contrast to room-scale pre-training, we see a line of work developing self-supervised methods tailored towards single object understanding tasks [19, 21, 27, 29, 34, 37, 40, 44, 49, 51]. They typically use the inherent structure and geometry of 3D point clouds to learn meaningful representations, *e.g.* by explicitly reconstructing point cloud patches using the Chamfer distance [34, 51], discriminating masked points from noise [29], or performing point cloud completion for occluded regions [44]. Another line of work additionally leverages multi-modal information to improve the latent representation of 3D point clouds, *i.e.* incorporating knowledge from models on 2D images [17, 21, 52, 53] or text descriptions [21, 52]. The advances of above methods are orthogonal to our approach point2vec as it operates on point clouds only. Most relevant to our work are Transformer-based self-supervised learning approaches on point clouds. Due to the sucess of pre-trained Transformer architectures in various domains [2, 9, 16, 23], we recently see a shift towards pre-training Transformer-based approaches for point clouds [29, 32, 49, 51]. Among them, Point-BERT [49] introduces a standard ViT-like [18] backbone to point clouds and extends BERT pre-training to point clouds [16]. Point-MAE [34] and Point-M2AE [51] follow the masked autoencoder approach proposed by He *et al.* [23]. In contrast to these methods, we do not explicitly reconstruct masked point cloud patches but predict contextualized targets in the latent feature space, circumventing the need to define sophisticated distance metrics to compare point cloud patches.

3 Method

The aim of this work is to unlock the full potential of data2vec-like [2] pre-training on point clouds by addressing point cloud specific challenges. To achieve this, we first summarize the technical concepts of data2vec (Sec. 3.1) and show how to learn rich representations on point clouds using data2vec pre-training (Sec. 3.2). Finally, we propose point2vec, which accounts for the point cloud specific limitations of data2vec (Sec. 3.3).

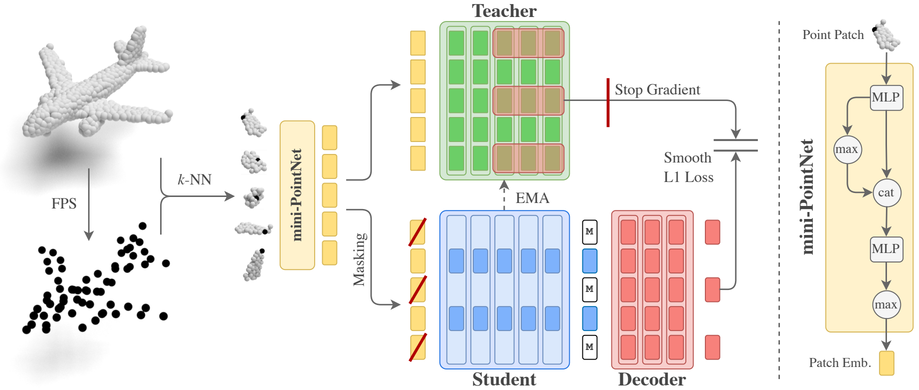


Fig. 2: Point2Vec pre-training. Our model divides the input point cloud into point patches using farthest point sampling (FPS) and k-NN aggregation. We obtain patch embeddings by applying a mini-PointNet 🟨 to each point patch (*right*). The teacher Transformer encoder 🟩 infers a contextualized representation for all patch embeddings which, after normalization and averaging over the last K Transformer layers, serve as training targets. The student's input is a masked view on the input data, *i.e.* we randomly mask out a ratio of patch embeddings and only pass the remaining embeddings into the student Transformer encoder 🟦. After applying a shallow decoder 🟥 on the outputs of the student, padded with learned mask embeddings [M], we train the student and decoder to predict the latent teacher representation of the patch embeddings.

3.1 Data2vec

Data2vec [2] is designed to pre-train Transformer-based models, which involve a feature encoder that maps the input data to a sequence of embeddings. These embeddings are subsequently passed to a standard Transformer encoder to generate the final latent representations. During pre-training, two versions of the Transformer encoder are kept: a *student* and a *teacher*. The teacher is a momentum encoder, *i.e.* its parameters $\mathit{\Delta}$ track the student's parameters θ by being updated after each training step according to an exponential moving average (EMA) rule [2, 9, 22, 24]: $\mathit{\Delta} \leftarrow \tau \mathit{\Delta} + (1 - \tau)\theta$, where $\tau \in [0, 1]$ is the EMA decay rate. The teacher provides the training targets, which the student predicts given a corrupted version of the same input. In a first step, the teacher encodes the uncorrupted input sequence. The training targets are then constructed by averaging the outputs of the last K blocks of the teacher, which are normalized to prevent a single block from dominating the sum. Due to the self-attention layers, these targets are *contextualized*, *i.e.* they incorporate global information from the whole input sequence. This is an important difference to other masked-prediction methods such as BERT [16] and MAE [23], where the targets only comprise local information, *e.g.* a word or an image patch. The student is given a masked version of the same input, where some of the embeddings in the input sequence are substituted by a special learned *mask embedding*. The student's task is to predict the targets corresponding to the masked parts of the input. The model is trained by optimizing a Smooth L1 loss on the regressed targets.

3.2 Data2vec for Point Clouds

To apply data2vec to point clouds, we utilize the same underlying model as Point-BERT [49] and Point-MAE [34]. This model is well suited for data2vec pre-training: it extracts a sequence of patch embeddings from the input point cloud and feeds it to a standard Transformer encoder. For downstream tasks, we append a task-specific head to the Transformer encoder (Sec. 4). Next, we describe the point cloud embedding and the Transformer in detail and conclude with a summary of data2vec for point clouds.

Point Cloud Embedding. First, we sample n center points from the input point cloud using farthest point sampling (FPS) [36]. Grouping the center points' k-nearest neighbors (k-NN) in the point cloud yields n contiguous *point patches*, *i.e.* sub-clouds of k elements. Next, we normalize the point patches by subtracting the corresponding center point from the patch's points. This untangles the positional and the structural information. As point clouds are permutation-invariant, we use a mini-PointNet [35] (Fig. 2, *right*) that maps each normalized point patch to a *patch embedding*.

The mini-PointNet involves the following steps: First, we map each point of a patch to a feature vector using a shared MLP. Then, we concatenate max-pooled features to each feature vector. The resulting feature vectors are then passed through a second shared MLP and a final max-pooling layer to obtain the patch embedding.

Transformer Encoder. The central component of the model is a standard Transformer encoder. The patch embeddings form the input sequence to the Transformer encoder. Since the point patches are normalized, the patch embeddings carry no positional information; therefore, a two-layer MLP maps each center point to a position embedding, which is then added to the corresponding patch embedding. Due to the special importance of positional information in point clouds, the position embeddings are added again before each subsequent Transformer block to ensure that the positional information is incorporated at every step of the encoding process.

Data2vec–pc. To establish a baseline, we apply the unmodified data2vec approach to the previously described underlying model of Point-BERT and Point-MAE. Going forward, we will refer to this approach as data2vec–pc.

3.3 Point2vec

In Fig. 2, we present the complete pipeline of our point2vec model. Directly applying data2vec to point cloud data without modifications is not optimal, as the position embeddings are also added to the mask embeddings, revealing the overall shape of the point cloud to the student. As positions are the only features, this makes the masking far less effective, as noted by Pang *et al.* [34] in the context of masked autoencoders.

To solve this issue, we adopt an approach inspired by MAE [23], where we only feed the non-masked embeddings to the student □. A separate decoder □, implemented as a shallow Transformer encoder, takes the output of the student and the previously held-back masked embeddings $\boxed{\texttt{M}}$ as input and predicts the training targets. In contrast to data2vec–pc, this approach does not suffer from leaking positional information from the masked-out point patches to the student. Moreover, utilizing an MAE-inspired setup provides additional benefits: First, the student is more computationally efficient, as it only needs to process the non-masked embeddings. Second, the model's inputs during fine-tuning are more similar to those during pre-training because they are no longer dominated by masked embeddings which are absent during fine-tuning. This likely makes the learned representations more transferable to downstream tasks.

4 Experiments

In this section, we describe the self-supervised pre-training of point2vec on ShapeNet [10] (Sec. 4.1). Next, we compare point2vec with top-performing self-supervised approaches and our baseline method data2vec–pc on three well-established datasets and four downstream tasks (Sec. 4.2). Finally, we put the spotlight on the architectural changes from our data2vec adaptation for point clouds to our proposed model point2vec which address the unique challenges of 3D point clouds (Sec. 4.3). In the supplementary material, we provide detailed hyperparameters of our model. Code and checkpoints will be made available.

4.1 Self-Supervised Pre-training

Following the pre-training protocol propagated by Point-BERT [49], Point-MAE [34] and Point-M2AE [51], we pre-train point2vec on the training split of ShapeNet [10] consisting of 41 952 synthetic 3D meshes of 55 categories, *e.g.* '*chair*', '*guitar*', '*airplane*'. We set the number of Transformer blocks to 12 with an internal dimension of 384. To pre-train our point-based approach, we uniformly sample 8192 points from the surfaces of the objects and then resample 1024 points using farthest point sampling [36]. During the point cloud embedding step we sample n=64 center points and k=32 nearest neighbors. We train point2vec with a batch size of 512 for 800 epochs using the AdamW [31] optimizer and a cosine learning rate decay [30] with a maximal learning rate of 10^{-3} after 80 epochs of linear warm-up. For data2vec–pc, we increase the batch size and learning rate to 2048 and 2×10^{-3}, respectively, as this empirically led to better results. Following data2vec [2], we set β=2 for the Smooth L1 loss and average the last K=6 blocks of the teacher. We use minimal data augmentations during pre-training: we randomly scale the input with a factor between $[0.8, 1.2]$ and rotate around the gravity axis. Pre-training takes roughly 18 hours on a single V100 GPU.

4.2 Main Results on Downstream Tasks

In order to evaluate the effectiveness of point2vec's self-supervised learning capabilities, we test point2vec against top-performing self-supervised methods on

Table 1: Part Segmentation on ShapeNetPart [48]. We report mean IoU across all part categories mIoU$_C$ and all instance mIoU$_I$.

Method	mIoU$_C$	mIoU$_I$
Transf.-OcCo [49]	83.4	85.1
Point-BERT [49]	84.1	85.6
MaskPoint [29]	84.4	86.0
Point-MAE [34]	84.1	86.1
Point-M2AE [51]	**84.9**	**86.5**
from scratch	84.1	85.7
data2vec–pc	84.1	85.9
point2vec (Ours)	84.6	86.3

Table 2: Shape Classification on ScanObjNN [43]. We report the overall accuracy over the three subsets `OBJ-BG`, `OBJ-ONLY` and the most challenging variant `PB-T50-RS`.

	Overall Accuracy					
Method	`OBJ-BG`		`OBJ-ONLY`		`PB-T50-RS`	
Transf.-OcCo [49]	84.9		85.5		78.8	
Point-BERT [49]	87.4		88.1		83.1	
MaskPoint [29]	89.3		89.7		84.6	
Point-MAE [34]	90.0		88.3		85.2	
Point-M2AE [51]	**91.2**		88.8		86.4	
from scratch	88.1	+1.6	88.8	−0.7	84.3	+1.2
data2vec–pc	89.7	+1.5	88.1	+2.3	85.5	+2.0
point2vec (Ours)	**91.2**		**90.4**		**87.5**	

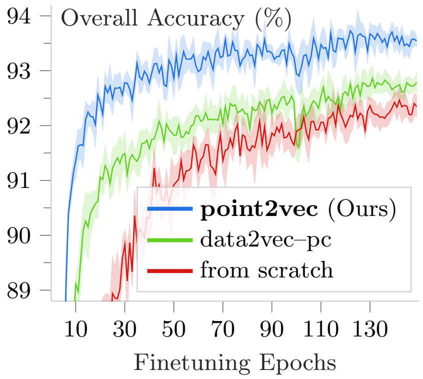


Fig. 3: Learning Curves for ModelNet40 [45]. We show the mean (solid line) and the standard deviation (shaded background) over 6 independent runs of point2vec, data2vec–pc as well as the model trained *from scratch* on ModelNet40. Point2vec consistently outperforms the baselines by a large margin.

	Overall Accuracy			
Method	+Voting		−Voting	
Transf.-OcCo [49]	92.1		–	
ParAE [19]	–		92.9	
STRL [27]	93.1		–	
Point-BERT [49]	93.2		92.7	
PointGLR [37]	–		93.0	
OcCo [44]	93.0		–	
MaskPoint [29]	93.8		–	
Point-MAE [34]	93.8		93.2	
Point-M2AE [51]	94.0		93.4	
from scratch	93.3	+0.3	93.0	+0.3
data2vec–pc	93.6	+1.2	93.3	+1.4
point2vec (Ours)	**94.8**		**94.7**	

Table 3: Shape Classification on ModelNet40 [45]. We report the overall accuracy with and without voting.

four different downstream tasks on well-established benchmarks. To that end, we discard the teacher network as well as the decoder and append a task-specific head to the student network. We then fine-tune the full network end-to-end for the specific task. We provide detailed hyperparameters for all downstream tasks in the supplementary material.

Synthetic Shape Classification. After pre-training on ShapeNet, we fine-tune our model for shape classification on ModelNet40 [45] consisting of 12 311 *synthetic* 3D models of 40 semantic categories. We obtain the semantic class label by passing the concatenated mean- and max-pooled output of the Transformer encoder into a 3-layer MLP and finetune the whole network end-to-end. We use minimal data augmentations consisting only of resampling 1024 points with farthest point sampling, applying random anisotropic scaling of up to 40%, centering at the origin, and rescaling to the unit sphere. Other commonly used augmentations did not improve performance, *e.g.* random rotations around the axis of gravity and random translations are detrimental as ModelNet40 instances are canonically oriented. During the point cloud embedding step we sample n=64 center points and k=32 nearest neighbors. In Tab. 3, we report a new state-of-the-art for shape classification on ModelNet40 [45] among self-supervised methods by a large margin of +1.3% without voting [34, 49, 51]. Interestingly, pre-training with data2vec–pc results only in marginal improvements (+0.3% without voting) over the same model trained *from scratch* on ModelNet40. Unlike data2vec–pc, we observe that point2vec unleashes the full potential of data2vec-like pre-training on ModelNet40 by achieving substantial performance gains of +1.7% over the baseline trained from scratch. In Fig. 3, we plot the accuracy per training epoch of point2vec, data2vec–pc, as well as our baseline trained *from scratch* on ModelNet40. We observe that point2vec outperforms our strong baselines by a consistent margin throughout the entire training. Point2vec effectively learns strong feature representations on ShapeNet, resulting in a significantly accelerated adaptation to the fine-tuning task (Fig. 3).

Real-World Shape Classification. Next, we fine-tune point2vec on ScanObjectNN [43] containing 2902 *real-world* object scans of 15 semantic classes. In contrast to shape classification on ModelNet40, we do not resample points but use all 2048 points and sample n=128 center points for the point cloud embedding step. We found more aggressive scaling to be detrimental and use random anisotropic scaling of up to 10%. Although pre-trained on synthetic data, Tab. 2 shows that point2vec generalizes well to cluttered real-world data and achieves state-of-the-art performance among self-super-vised methods by a significant margin of +1.1% on `PB-T50-RS`, the most difficult variant of the dataset. We observe that pre-training point2vec on ShapeNet plays a crucial role to its strong performance. Compared to the baseline trained from scratch on ScanObjectNN, pre-training with point2vec achieves an performance gain of +3.2%. We again report improvements of point2vec over data2vec–pc of up to +2.3%.

Few-Shot Classification. Following the standard evaluation protocol proposed by Sharma *et al.* [40], we test the few-shot capabilities of point2vec in a m-way, n-shot setting. To this end, we randomly sample m classes and select n instances for training at random for each of these classes. For testing, we randomly pick 20 unseen instances from each of the m support classes. We provide the standard deviation over 10 independent runs. In Tab. 4, we report a new state-of-the-art by improvements up to +1.3% in the most difficult 10-way 10-shot setting. Point2vec clearly outperforms the data2vec–pc baseline in all settings. We conclude that point2vec learns rich feature representations which are also well suited for transfer learning in a low-data regime.

Table 4: Few-Shot Classification on ModelNet40 [45]. Mean and standard deviation over 10 runs.

	5-way		10-way	
Method	10-shot	20-shot	10-shot	20-shot
OcCo [44]	91.9±3.6	93.9±3.1	86.4±5.4	91.3±4.6
Transf.-OcCo [49]	94.0±3.6	95.9±2.3	89.4±5.1	92.4±4.6
Point-BERT [49]	94.6±3.1	96.3±2.7	91.0±5.4	92.7±5.1
MaskPoint [29]	95.0±3.7	97.2±1.7	91.4±4.0	93.4±3.5
Point-MAE [34]	96.3±2.5	97.8±1.8	92.6±4.1	95.0±3.0
Point-M2AE [51]	96.8±1.8	98.3±1.4	92.3±4.5	95.0±3.0
from scratch	93.8±3.2	97.1±1.9	90.1±4.6	93.6±3.9
data2vec–pc	96.2±2.6	97.8±2.2	92.6±4.9	95.0±3.2
point2vec (Ours)	**97.0**±2.8	**98.7**±1.2	**93.9**±4.1	**95.8**±3.1

Part Segmentation. Finally, we address the task of part segmentation, which assigns a semantic part label to each point in a 3D point cloud of a single object. For this purpose, we employ a simple segmentation head that is similar to the segmentation head in Point-MAE [34]. First, we average the outputs of the 4th, 8th, and 12th Transformer blocks to incorporate features from multiple levels of abstraction. We then concatenate the mean- and max-pooling of the n averaged token outputs, along with the one-hot encoded class label of the object, to obtain a global feature vector. At the same time, we up-sample the n averaged outputs from the corresponding center points to all points using a PointNet++ [36] *feature propagation layer*, which uses inverse distance weighting and a shared MLP to produce a local feature vector for each point. Finally, we concatenate the global feature vector with each local feature vector and a shared MLP predicts a part label for each point. In Tab. 1, we report competitive results on ShapeNetPart [48] which consists of 16 881 3D models of 16 semantic categories. Apart from Point-M2AE [51], point2vec outperforms all other self-supervised methods. We hypothesize that Point-M2AE's multi-scale U-Net like architecture [38] enables to learn more expressive spatially localized features which results in slightly better scores (+0.2 mIoU$_I$). Since point2vec relies on a standard single-scale Transformer backbone, we see multi-scale Transformers for 3D point clouds as an interesting orthogonal improvement, similar to the advances in 2D vision [11, 20, 28, 50] extending vision Transformers [18] with multi-scale capabilities.

4.3 Analysis

Leakage of Positional Information. The main limitation of data2vec–pc is that it directly feeds masked embeddings, along with their positional information, to the student network, which undermines the effectiveness of masking. To visualize this problem, we show a representative example in Fig. 4(a). Re-

Table 5: Ablation. We find that a deferred shallow decoder (**D**) (Fig. 2 □) predicting the teacher's representations for masked patches shows consistent improvements but we identify that concealing positional information (**no** 🄼) from the student is key.

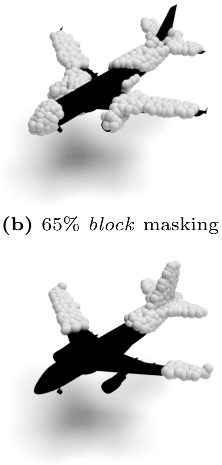
			Overall Accuracy		
			ModelNet40		**ScanObjNN**
	no 🄼	D	+Voting	−Voting	`PB-T50-RS`
data2vec–pc	✗	✗	93.6	93.3	85.5
	✗	✓	94.0	93.6	86.8
point2vec	✓	✓	**94.8**	**94.7**	**87.5**

(a) Disclosed Positions (**data2vec–pc**) **(b)** Concealed Positions (**point2vec**)

Fig. 4: Leakage Of Positional Information. The center points ● of masked point patches are associated with the masked embeddings (Fig. 2, 🄼). **(a)** data2vec–pc discloses the positions of masked patches to the student, revealing the chair's overall shape. **(b)** point2vec excludes masked embeddings from the student and therefore conceals the positions of the masked patches.

vealing the positions of masked patches of the chair inadvertently weakens the learning objective because it allows the student to rely on the positional information instead of truly learning to predict the teacher's representations of the corresponding masked-out patches. To mitigate this issue, point2vec excludes masked embeddings from the student and only subsequently feeds them to the decoder. As a result, several sections of the chair in Fig. 4(b) are effectively concealed from the student network, leading to a more resilient learning framework. In Tab. 5, we report that point2vec outperforms our baseline data2vec–pc by a significant margin of up to +2.0%. In particular, we observe that the decoder itself provides consistent improvements, but the key contribution of point2vec is to conceal positional information from the student, *i.e.*, shifting mask tokens from the encoder's input to the decoder (**no** 🄼). Complementary to our findings, He *et al.* [23] show that moving masked embeddings to a deferred shallow decoder reduces memory requirements and training time significantly. Our findings align with those of Pang *et al.* [34], who found similar benefits for masked autoencoders on point clouds.

Table 6: Masking Strategy. We explore two variants for masking the input of the student. For **(a)** random masking, we uniformly mask out a given ratio of all embeddings. For **(b)** block masking, we mask out a random embedding and its nearest neighbors. We report the overall accuracy on ModelNet40 and ScanObjectNN.


(a) 65% *random* masking

(b) 65% *block* masking

		Overall Accuracy		
		ModelNet40		**ScanObjNN**
Strategy	Masking Ratio	+Voting	−Voting	PB-T50-RS
random	45%	94.5	94.3	86.8
random	65%	**94.8**	**94.7**	**87.5**
random	85%	94.5	93.8	86.7
block	25%	93.9	93.7	86.3
block	45%	94.5	93.8	87.4
block	65%	94.0	93.9	86.1

Masking Strategy. The masking strategy defines which of the student's input embeddings are masked (Fig. 2, /). In this study, we investigate two variants of masking strategies with different masking ratios: *random* masking and *block* masking. For random masking, we mask out a specified ratio of embeddings for the student. In contrast, block masking masks out a random embedding and its nearest neighbors such that the specified masking ratio is achieved. This strategy puts focus on masking out spatially contiguous regions of the point cloud whereas random masking is independent of position. Our findings, summarized in Tab. 6, reveal that random masking with a 65% masking ratio performs best for both ModelNet40 and ScanObjectNN, while block masking lags behind. We attribute this to the high level of ambiguity that arises when masking a spatially contiguous region, resulting in several potential point clouds that could have given rise to the masked input. While we seek a challenging pretext task to learn rich representations, ambiguity should not be the primary source of difficulty.

We recap that our masking strategy is applied to patch embeddings rather than individual points. Consequently, points may belong to *both* masked and unmasked patches. While certain masked patches may be easy to predict, our masking ratio of 65% ensures that there are still plenty of regions entirely masked (Fig. 4, Tab. 6). As a result, we conclude that the masking ratio is a sensitive hyperparameter that requires some careful tuning to strike the right balance of difficulty for the pretext task.

Visualization of representations learned by point2vec. In Fig. 5, we show qualitative examples of representations of ModelNet40 instances after pre-training on ShapeNet. Both a random initialization and data2vec–pc pre-training show a

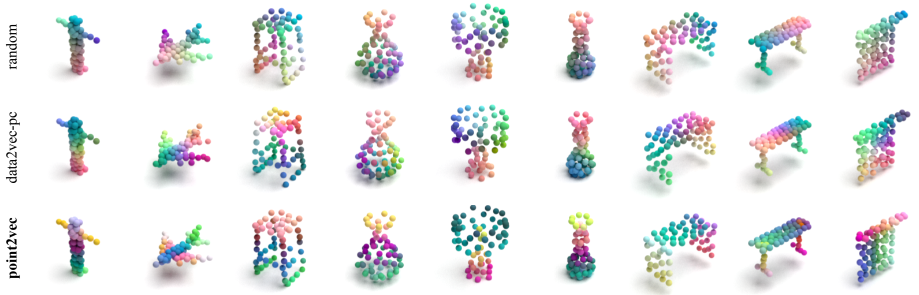


Fig. 5: Visualization of Learned Representations. We use PCA to project the learned representations into RGB space. Both a random initialization and data2vec–pc pre-training show a fairly strong positional bias, whereas point2vec exhibits a stronger semantic grouping without being trained on downstream dense prediction tasks.

strong positional bias, whereas point2vec exhibits a stronger semantic grouping without being trained on downstream dense prediction tasks. Unlike data2vec–pc, point2vec conceals positional information from the student, forcing it to learn more about the semantics of the data, resulting in more semantically meaningful representations.

5 Conclusion

In this work, we have extended data2vec to the point cloud domain. Through an in-depth analysis, we have discovered that the disclosure of positional information to the student network hampers data2vec's ability to learn strong representations on point clouds. To overcome this limitation, we have introduced point2vec, a self-supervised representation learning approach which unleashes the full potential of data2vec-like pre-training on point clouds. Point2vec achieves remarkable results on various downstream tasks, surpassing other self-supervised learning approaches in few-shot learning as well as shape classification on well-established benchmarks. Future work might include extending point2vec for scene-level representation learning.

Acknowledgments. This work has been funded, in parts, by the ERC CoG grant DeeViSe (ERC-2017-CoG-773161) and by the BMBF project 6GEM (16KISK036K). We gratefully acknowledge computing resources granted by RWTH Aachen University (thes1313, supp0003) and thank Idil Esen Zulfikar and István Sárándi for helpful discussions and feedback.

References

1. Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S.: 3D Semantic Parsing of Large-Scale Indoor Spaces. In: IEEE Conference on Computer Vision and Pattern Recognition (2016)
2. Baevski, A., Hsu, W.N., Xu, Q., Babu, A., Gu, J., Auli, M.: Data2vec: A general framework for self-supervised learning in speech, vision and language. In: International Conference on Machine Learning (2022)
3. Baevski, A., Zhou, Y., Mohamed, A., Auli, M.: wav2vec 2.0: A framework for self-supervised learning of speech representations. In: Neural Information Processing Systems (2020)
4. Bao, H., Dong, L., Piao, S., Wei, F.: BEiT: BERT pre-training of image transformers. In: International Conference on Learning Representations (2022)
5. Bardes, A., Ponce, J., LeCun, Y.: Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In: International Conference on Learning Representations (2022)
6. Baruch, G., Chen, Z., Dehghan, A., Dimry, T., Feigin, Y., Fu, P., Gebauer, T., Joffe, B., Kurz, D., Schwartz, A., Shulman, E.: ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In: Neural Information Processing Systems (2021)
7. Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J.: SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In: International Conference on Computer Vision (2019)
8. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuScenes: A multimodal dataset for autonomous driving. In: IEEE Conference on Computer Vision and Pattern Recognition (2020)
9. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: International Conference on Computer Vision (2021)
10. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
11. Chen, C.F.R., Fan, Q., Panda, R.: CrossViT: Cross-Attention Multi-Scale Vision Transformer for Image Classification. In: International Conference on Computer Vision (2021)
12. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A Simple Framework for Contrastive Learning of Visual Representations. In: International Conference on Machine Learning (2020)
13. Chen, X., He, K.: Exploring Simple Siamese Representation Learning. In: IEEE Conference on Computer Vision and Pattern Recognition (2021)
14. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (2019)
15. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2017)
16. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. "Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (2018)

17. Dong, R., Qi, Z., Zhang, L., Zhang, J., Sun, J., Ge, Z., Yi, L., Ma, K.: Autoencoders as Cross-Modal Teachers: Can Pretrained 2D Image Transformers Help 3D Representation Learning? In: International Conference on Learning Representations (2023)
18. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2020)
19. Eckart, B., Yuan, W., Liu, C., Kautz, J.: Self-Supervised Learning on 3D Point Clouds by Learning Discrete Generative Models. In: IEEE Conference on Computer Vision and Pattern Recognition (2021)
20. Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale Vision Transformers. In: International Conference on Computer Vision (2021)
21. Gao, M., Xing, C., Martín-Martín, R., Wu, J., Xiong, C., Xue, L., Xu, R., Niebles, J.C., Savarese, S.: Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In: IEEE Conference on Computer Vision and Pattern Recognition (2023)
22. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al.: Bootstrap your own latent-a new approach to self-supervised learning. In: Neural Information Processing Systems (2020)
23. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: IEEE Conference on Computer Vision and Pattern Recognition (2022)
24. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: IEEE Conference on Computer Vision and Pattern Recognition (2020)
25. Hou, J., Graham, B., Nießner, M., Xie, S.: Exploring data-efficient 3d scene understanding with contrastive scene contexts. In: IEEE Conference on Computer Vision and Pattern Recognition (2021)
26. Hsu, W.N., Tsai, Y.H.H., Bolte, B., Salakhutdinov, R., Mohamed, A.: Hubert: How Much Can a Bad Teacher Benefit ASR Pre-Training? In: IEEE Conference on Acoustics, Speech and Signal Processing (2021)
27. Huang, S., Xie, Y., Zhu, S.C., Zhu, Y.: Spatio-temporal Self-Supervised Representation Learning for 3D Point Clouds. In: International Conference on Computer Vision (2021)
28. Li, Y., Wu, C.Y., Fan, H., Mangalam, K., Xiong, B., Malik, J., Feichtenhofer, C.: MViTv2: Improved multiscale vision transformers for classification and detection. In: IEEE Conference on Computer Vision and Pattern Recognition (2022)
29. Liu, H., Cai, M., Lee, Y.J.: Masked Discrimination for Self-Supervised Learning on Point Clouds. In: European Conference on Computer Vision (2022)
30. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. In: International Conference on Learning Representations (2017)
31. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2019)
32. Ma, X., Qin, C., You, H., Ran, H., Fu, Y.: Rethinking network design and local geometry in point cloud: A simple residual mlp framework. In: International Conference on Learning Representations (2022)
33. van den Oord, A., Li, Y., Vinyals, O.: Representation Learning with Contrastive Predictive Coding. arXiv preprint arXiv:1807.03748 (2018)

34. Pang, Y., Wang, W., Tay, F.E., Liu, W., Tian, Y., Yuan, L.: Masked autoencoders for point cloud self-supervised learning. In: European Conference on Computer Vision (2022)
35. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (2017)
36. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In: Advances in Neural Information Processing Systems (2017)
37. Rao, Y., Lu, J., Zhou, J.: Global-Local Bidirectional Reasoning for Unsupervised Representation Learning of 3D Point Clouds. In: IEEE Conference on Computer Vision and Pattern Recognition (2020)
38. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention (2015)
39. Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B.: Mask3D for 3D Semantic Instance Segmentation. In: IEEE International Conference on Robotics and Automation (2023)
40. Sharma, C., Kaul, M.: Self-Supervised Few-Shot Learning on Point Clouds. In: Neural Information Processing Systems (2020)
41. Tian, Y., Krishnan, D., Isola, P.: Contrastive Multiview Coding. In: European Conference on Computer Vision (2020)
42. Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., Isola, P.: What Makes for Good Views for Contrastive Learning? In: Neural Information Processing Systems (2020)
43. Uy, M.A., Pham, Q.H., Hua, B.S., Nguyen, D.T., Yeung, S.K.: Revisiting Point Cloud Classification: A New Benchmark Dataset and Classification Model on Real-World Data. In: International Conference on Computer Vision (2019)
44. Wang, H., Liu, Q., Yue, X., Lasenby, J., Kusner, M.J.: Unsupervised Point Cloud Pre-Training via Occlusion Completion. In: International Conference on Computer Vision (2021)
45. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d ShapeNets: A Deep Representation for Volumetric Shapes. In: International Conference on Computer Vision (2015)
46. Xie, S., Gu, J., Guo, D., Qi, C.R., Guibas, L., Litany, O.: PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding. In: European Conference on Computer Vision (2020)
47. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: XLNet: Generalized Autoregressive Pretraining for Language Understanding. In: Neural Information Processing Systems (2019)
48. Yi, L., Kim, V.G., Ceylan, D., Shen, I.C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.: A Scalable Active Framework for Region Annotation in 3D Shape Collections. SIGGRAPH Asia (2016)
49. Yu, X., Tang, L., Rao, Y., Huang, T., Zhou, J., Lu, J.: Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In: IEEE Conference on Computer Vision and Pattern Recognition (2022)
50. Zhang, P., Dai, X., Yang, J., Xiao, B., Yuan, L., Zhang, L., Gao, J.: Multi-Scale Vision Longformer: A New Vision Transformer for High-Resolution Image Encoding. In: International Conference on Computer Vision (2021)

51. Zhang, R., Guo, Z., Gao, P., Fang, R., Zhao, B., Wang, D., Qiao, Y., Li, H.: Point-M2AE: Multi-scale masked autoencoders for hierarchical point cloud pre-training. In: Advances in Neural Information Processing Systems (2022)
52. Zhang, R., Guo, Z., Zhang, W., Li, K., Miao, X., Cui, B., Qiao, Y., Gao, P., Li, H.: Pointclip: Point cloud understanding by clip. In: IEEE Conference on Computer Vision and Pattern Recognition (2022)
53. Zhang, R., Wang, L., Qiao, Y., Gao, P., Li, H.: Learning 3D Representations from 2D Pre-trained Models via Image-to-Point Masked Autoencoders. In: IEEE Conference on Computer Vision and Pattern Recognition (2023)