Temporal Consistent 3D LiDAR Representation Learning for Semantic Perception in Autonomous Driving

Lucas Nunes[*] Louis Wiesmann[*] Rodrigo Marcuzzi[*]
Xieyuanli Chen[*] Jens Behley[*] Cyrill Stachniss[*,⋆,‡]
[*]University of Bonn, [⋆]University of Oxford, [‡]Lamarr Institute
{firstname.lastname}@igg.uni-bonn.de

Abstract

Semantic perception is a core building block in autonomous driving, since it provides information about the drivable space and location of other traffic participants. For learning-based perception, often a large amount of diverse training data is necessary to achieve high performance. Data labeling is usually a bottleneck for developing such methods, especially for dense prediction tasks, e.g., semantic segmentation or panoptic segmentation. For 3D LiDAR data, the annotation process demands even more effort than for images. Especially in autonomous driving, point clouds are sparse, and objects appearance depends on its distance from the sensor, making it harder to acquire large amounts of labeled training data. This paper aims at taking an alternative path proposing a self-supervised representation learning method for 3D LiDAR data. Our approach exploits the vehicle motion to match objects across time viewed in different scans. We then train a model to maximize the point-wise feature similarities from points of the associated object in different scans, which enables to learn a consistent representation across time. The experimental results show that our approach performs better than previous state-of-the-art self-supervised representation learning methods when fine-tuning to different downstream tasks. We furthermore show that with only 10% *of labeled data, a network pre-trained with our approach can achieve better performance than the same network trained from scratch with all labels for semantic segmentation on SemanticKITTI.* [1]

1. Introduction

Semantic perception is essential for safe interaction between autonomous vehicles and their surrounding. For learning-based perception, a massive amount of training data is usually required for training high-performance models. However, the data annotation is the bottleneck of collecting such large training sets, especially for dense prediction tasks, such as semantic segmentation [60, 80], instance segmentation [46, 73], and panoptic segmentation [45, 79], which require fine-grained labels. In the context of autonomous driving, recent approaches exploit 3D LiDAR data [20, 54, 55, 61], where the data annotation process is more complex than on 2D RGB images due to the sparsity of the point cloud and object appearance varying with its distance to the sensor.

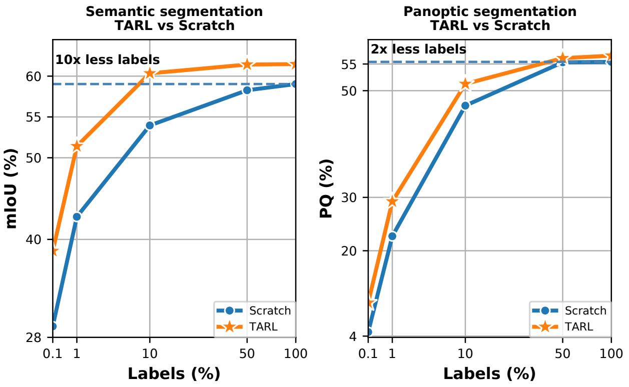

Figure 1. Our pre-training (TARL) can reduce the amount of necessary labels during fine-tuning on SemanticKITTI [4, 22]. Our method requires only 10% of labels to surpass the network trained from scratch with the full dataset for semantic segmentation. For panoptic segmentation, our method requires only half of the labels.

Recent self-supervised representation learning methods [10, 11, 13–15, 25, 27, 76] tackle the lack of annotated data with a pre-training step requiring no labels. Those methods use data augmentation to generate different views from one data sample and train the network to learn an embedding space able to have similar representations for the generated augmented views. Other approaches [52, 63, 65, 69, 78] propose optimizing the pixel embedding space to learn a dense representation suited to be fine-tuned to more fine-grained tasks. For 3D point cloud data, recent approaches [21, 42, 49, 56] focus on synthetic point clouds of single objects to learn a robust representation for object classification. Other approaches [31, 36, 50, 67, 74, 75] target real-world data representation, such as LiDAR or RGB-D data, but fewer target autonomous driving scenarios.

[1] Code: https://github.com/PRBonn/TARL

In this paper, we propose a novel temporal association representation learning (TARL) method for 3D LiDAR data designed for autonomous driving data. We exploit the vehicle motion to extract different views of the same objects across time. Then, we compute point-wise features from the objects in the point cloud and use a Transformer encoder [2] as a projection head to learn a temporal association from the object representation, embedding the objects dynamics. We conduct extensive experiments to compare our approach with state-of-the-art methods and show that our approach surpasses previous self-supervised pre-training methods [50, 67, 75] when fine-tuning to different downstream tasks and datasets [4, 8, 22, 64]. In summary, our key contributions are:

- We propose a novel self-supervised pre-training for 3D LiDAR data able to learn a robust and temporally-consistent representation by clustering together points from the same object viewed at different points in time.
- We achieve better performance than previous state-of-the-art methods on different downstream tasks, *i.e.*, semantic segmentation, panoptic segmentation, and object detection.
- With our pre-training, we require only 10% of labels to surpass the network trained from scratch for semantic segmentation using the whole training set on SemanticKITTI (see Fig. 1).
- Our self-supervised pre-training produces representations more suited for transfer learning than supervised pre-training, achieving better performance when fine-tuning to a different dataset.

2. Related Work

Self-supervised representation learning aims at initializing weights of a deep neural network for a downstream task by learning suitable representations *without* labeled data. Early approaches [18, 26, 28, 33, 48, 70] use so-called pretext tasks to learn useful representations, such as solving jigsaw puzzles [28, 33, 48] or reconstructing masked data [18, 26, 70]. More recently, contrastive learning [13, 14, 27] has drawn significant attention in the vision and robotics research community. Those contrastive methods employ random data augmentation over one image sample to generate different views of it. Then, the network is trained to learn an embedding space able to discriminate between the augmented samples from the same image and the augmented views from other images. Recent approaches exploit this discriminative pretext task and propose different ways to learn a self-supervised data representation, *e.g.*, via non-contrastive methods [15, 25], redundancy reduction [76], or online clustering [10]. Other works [52,63,65,69,78] approach this in a pixel-wise manner, learning dense representations applied to more fine-grained downstream tasks.

Transformer encoders [2] were initially proposed in the natural language processing community achieving state-of-the-art performance in the field. Such architectures were then quickly adopted by the computer vision community [1, 30, 66, 72, 77]. With the attention mechanism, the Transformer can learn the global context from the image instead of just local neighboring context as in convolutional models [1]. With the growing interest in Transformers, some self-supervised representation learning methods were applied to Transformers [11,59,68] to evaluate the capacity of such models to learn meaningful representations.

LiDAR point cloud perception brings further challenges compared to the image domain. Previous works [46, 47] deal with 3D scans by projecting the data to a 2D range image and then processing it with 2D convolutional methods. More recently, different convolution architectures were proposed to deal with 3D data, such as point kernels [54, 55, 61], sparse convolutions [20, 23, 24], or graph representations [37, 40, 62]. With those different architectures, it becomes possible to train a model to learn from 3D data and deal with different tasks in this domain, such as semantic segmentation [60,80], panoptic segmentation [3,29,44], object detection [38,57,58], or moving object segmentation [6, 16]. With increasing interest in Transformers, some works [12,19,34,70] proposed ways of dealing with point cloud data in such architectures. In this domain, a key challenge lies in the number of points collected by LiDAR sensors together with the high memory requirements from the Transformer attention mechanism.

3D self-supervised representation learning aims at learning robust representations from point cloud data. Most of those methods focus on point clouds of single objects [21, 42,49,56]. Other methods [31,36,50,67,74,75] target dense point-wise representations for real-world point cloud scans. Xie *et al.* [67] propose learning a representation via point-to-point contrastive learning through matching corresponding points in two augmented point clouds. Similarly, Hou *et al.* [31] learn both point-wise correspondence and the region where the point is located in the point cloud. Zhang *et al.* [75] propose global representation learning through scan-to-scan discrimination. Chen *et al.* [74] synthetically add dynamic objects in the scene to embed temporal and dynamic understanding in the learned representation. Other methods propose discriminating regions of interest to extract more semantic information. Nunes *et al.* [50] extract coarse segments from point clouds and use contrastive loss to distinguish between objects segments. Similarly, Yin *et al.* [36] extract proposal regions to be discriminated, biasing the pre-training to object detection.

Despite the performance improvement brought by those 3D self-supervised representation learning methods, there is

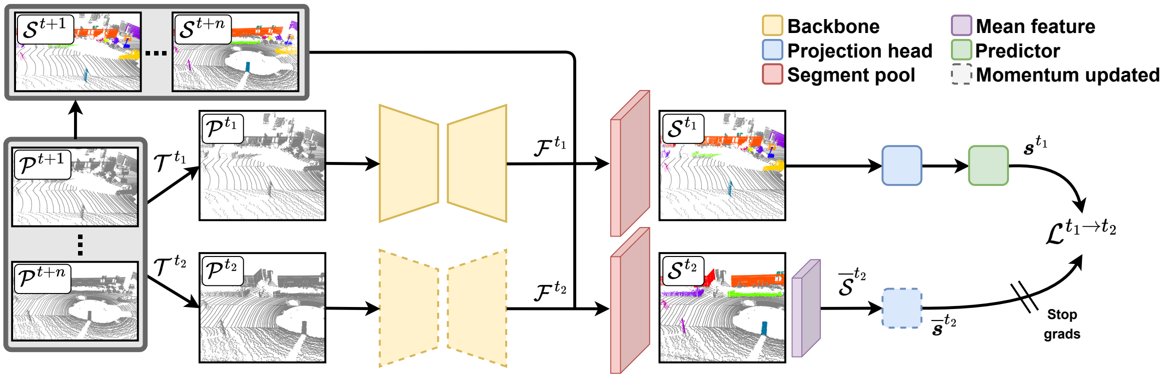

Figure 2. We aggregate n point clouds and extract segments $\mathcal{S}$ from it by removing the ground and clustering the remaining points. Then we sample two point clouds $\mathcal{P}^{t_1}$ and $\mathcal{P}^{t_2}$, and apply random augmentations $\mathcal{T}^{t_1}$ and $\mathcal{T}^{t_2}$ to them. Next, we compute point-wise features for each scan and list its segments $\mathcal{S}^{t_1}$ and $\mathcal{S}^{t_2}$ with their corresponding point features. For $\mathcal{S}^{t_2}$ we compute the mean feature vector for each segment and compute the segments target embeddings $\overline{\boldsymbol{s}}^{t_2}$ with a projection head. For $\mathcal{S}^{t_1}$ we use a point-wise projector followed by a predictor to identify for each segment point features $\boldsymbol{s}^{t_1}_{m,p}$ the corresponding target embedding $\overline{\boldsymbol{s}}^{t_2}_{m}$ by minimizing the loss $\mathcal{L}^{t_1 \to t_2}$.

still a performance gap compared to the image domain due to the lack of data augmentations for point clouds. Similar to previous works [36,50], our approach focuses on learning representations over regions of interest. However, unlike previous methods, we extract natural augmented views of the objects in the scans collected at different points in time while the vehicle drives in the environment. Then, the network is trained to learn a common representation for those temporal views of one object. This enables our method to learn a more robust semantic representation by learning temporal consistent features and real-world dynamics.

3. Method

In this paper, we propose a new self-supervised representation learning method for LiDAR data obtained in the context of autonomous driving. Our approach requires only unlabeled point clouds and the corresponding scan pose. Pose information is usually readily available by means of GPS/IMU, odometry approaches [32, 53], or SLAM systems [5, 17], thus, this is not a limitation in practice. We use a siamese network scheme with an online updated network and a momentum updated network (see Fig. 2). The whole pipeline consists of extracting objects as coarse segments viewed at different times over an interval of scans. Then, point-wise features are computed with the backbone and a Transformer encoder as a projector. Finally, we perform an implicit clustering to put points from different views of the same object together. In the following subsections, we explain the individual steps of our method thoroughly.

3.1. Temporal objects views

Instead of only using data augmentation to generate artificial views of one object, we exploit the vehicle motion to extract real-world object segments viewed from different perspectives. Given the vehicle motion and the properties of LiDAR sensors, one object can have different appearances depending on its position relative to the sensor. Such changing appearance is usually a problem when fine-tuning the model to data collected with different sensors [7, 35, 39, 71]. However, we exploit these properties in our favor to extract natural augmented views of one object. Previous works [31, 36, 50, 67, 74, 75] relied only on data augmentation to generate pairs of one scan, such as rotation, translation and scaling. Instead, we extract object views collected from different perspectives by the LiDAR sensor while the vehicle navigates through the environment. We then apply augmentations over the associated scans captured at a different time and train the network to learn a common representation for the object views.

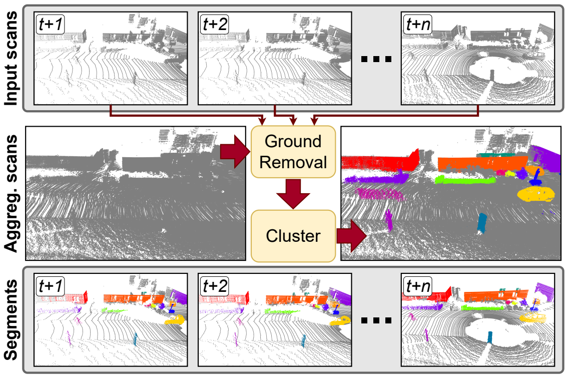

Figure 3. We aggregate n point clouds from different times and extract coarse segments from the aggregated point cloud in an unsupervised manner. We keep the point indices mapping to extract views at different times from one object as augmented pairs.

We define a point cloud at time t as $\mathcal{P}^t = \{\boldsymbol{p}^t_1, ..., \boldsymbol{p}^t_R\}$ as a set of 3D points $\boldsymbol{p}_r \in \mathbb{R}^3$. To extract coarse object segments in an individual scan, we first separate $\mathcal{P}^t$ into ground $\mathcal{G}^t$ and non-ground points $\hat{\mathcal{P}}^t$ in an unsupervised manner using the method proposed by Lim *et al.* [41], where $\mathcal{P}^t = \hat{\mathcal{P}}^t \cup \mathcal{G}^t$ and $\hat{\mathcal{P}}^t \cap \mathcal{G}^t = \emptyset$. Af-

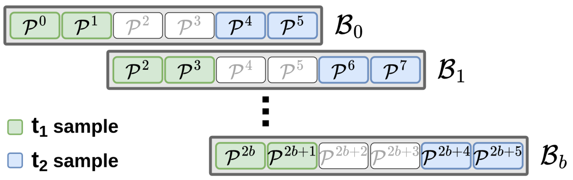
$\mathcal{P}^0$ $\mathcal{P}^1$ $\mathcal{P}^2$ $\mathcal{P}^3$ $\mathcal{P}^4$ $\mathcal{P}^5$
$\mathcal{B}_0$
$\mathcal{P}^2$ $\mathcal{P}^3$ $\mathcal{P}^4$ $\mathcal{P}^5$ $\mathcal{P}^6$ $\mathcal{P}^7$
$\mathcal{B}_1$
🟩 t_1 sample
🟦 t_2 sample
$\mathcal{P}^{2b}$ $\mathcal{P}^{2b+1}$ $\mathcal{P}^{2b+2}$ $\mathcal{P}^{2b+3}$ $\mathcal{P}^{2b+4}$ $\mathcal{P}^{2b+5}$
$\mathcal{B}_b$

Figure 4. Batch example, for an interval of $n = 6$ point clouds $\mathcal{P}$. We divide a sequence of N scans into batches $\mathcal{B}_b$ where we sample t_1 from the beginning and t_2 from the end of the interval. Then, the next batch starts in the middle of the previous interval.

terwards, we cluster the non-ground points $\hat{\mathcal{P}}^t$ into M segments $\mathcal{S}^t = \{\mathcal{S}_1^t, ..., \mathcal{S}_M^t\}$ using HDBSCAN [9], where $\hat{\mathcal{P}}^t = \bigcup_{m=1}^{M} \mathcal{S}_m^t$.

To map different segment views at different times, we define an interval of n scans from which the views of the object will be extracted. Those scans are transformed to a common global coordinate frame to be then aggregated. The global scan poses can be easily acquired with GPS/IMU, SLAM systems [5, 17], or LiDAR odometry [32, 53]. The aggregated point cloud $\mathcal{P}$ is given by $\mathcal{P} = \{\mathcal{P}^{t+1}, \mathcal{P}^{t+2}, ..., \mathcal{P}^{t+n}\}$. Similarly, we can aggregate the individual ground segmentation labels $\mathcal{G} = \{\mathcal{G}^{t+1}, \mathcal{G}^{t+2}, ..., \mathcal{G}^{t+n}\}$ and get the aggregated non-ground points $\hat{\mathcal{P}} = \{\hat{\mathcal{P}}^{t+1}, \hat{\mathcal{P}}^{t+2}, ..., \hat{\mathcal{P}}^{t+n}\}$. As in the individual scan case, we can cluster $\hat{\mathcal{P}}$ to get the M segments $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_M\}$. By keeping the point index mapping from the aggregated point cloud to the individual n scans, we can identify the n segments of the same object viewed at different times as $\mathcal{S}_m = \{\mathcal{S}_m^{t+1}, ..., \mathcal{S}_m^{t+n}\}$. We then list the temporal views from each of the M segments as $\mathcal{S}_{1:M} = \{\mathcal{S}_1^{t+1}, ..., \mathcal{S}_1^{t+n}, ..., \mathcal{S}_M^{t+1}, ..., \mathcal{S}_M^{t+n}\}$.

To properly segment views of both static and moving objects, we assume a LiDAR at the commonly used frequency of 10 Hz, where the scan overlap is enough to cluster together also moving objects, as depicted in Fig. 3. We show further examples in the supplementary material. With this procedure, our augmented pairs are views of the same object at different times from static and moving objects. By maximizing the similarity between the object views at different times, the network needs to learn a more general representation consistent across time and embed the object dynamics. In the supplementary material, we provide an experimental comparison between the pre-training using the temporal object views as augmented pairs and with augmented pairs generated only with data augmentation to validate the use of temporal views.

3.2. Temporal batch

During pre-training, we need to sample a subset of n consecutive scans from the training sequence to be aggregated and extract the temporal object views as explained in Sec. 3.1. This sample will be used during pre-training to learn a temporal consistent representation for the segmented objects. For a sequence of N point clouds, we divide it into batches $\mathcal{B}$ with intervals of n scans. During the pre-training forward pass, we sample two scans at different times, t_1 and t_2. Since the goal of this temporal sampling is to have different views of the same object, we enforce this by sampling t_1 from the beginning and t_2 from the end of this n scans interval such that $t_1 < \frac{n}{3}$ and $t_2 \geq 2\frac{n}{3}$. In this case, the scans between $\frac{n}{3} \leq t < 2\frac{n}{3}$ would never be sampled. To also process those scans, the next batch starts at this unseen interval. Therefore, for a batch sample $\mathcal{B}_b$ the n scans from the sequence of N point clouds are $\mathcal{B}_b = \{\mathcal{P}^k | b\frac{n}{3} \leq k < b\frac{n}{3} + n\}$.

In Figure 4, we show how the batches would look like in an example with $n = 6$ scans interval. Even though we do not sample t from $\frac{n}{3} \leq t < 2\frac{n}{3}$, we still need to aggregate and cluster these scans to correctly segment dynamic objects and match their correspondences between the scans.

With this sampling scheme, we guarantee that the objects views will be distant in time and still all the data will be seen during pre-training.

3.3. Implicit clustering

With our coarse object segments $\mathcal{S}$, we have the prior that points from the same segment should be semantically similar. Given that we rely on a set of object segments, one straightforward way to distinguish the learned embeddings is to cluster together point features from the same object. Recent methods propose an online clustering scheme to separate samples around a fixed number of prototype learnable cluster centers [10]. We aim at clustering point features from the same object close together but in a more straightforward way with an implicit clustering.

Given the temporal sampled pair $\mathcal{P}^{t_1}$ and $\mathcal{P}^{t_2}$ from a batch sample $\mathcal{B}_b$, we compute point-wise features $\mathcal{F}^{t_1}$ and $\mathcal{F}^{t_2}$ with the backbone. As the target embedding, we list from $\mathcal{F}^{t_2}$ the set of M segments $\mathcal{S}^{t_2}$. We compute for each segment a mean representation from its point-wise features, and project this embedding with a self-attention Transformer encoder as a projection head to get the M target mean feature vectors $\bar{\mathbf{s}}^{t_2} \in \mathbb{R}^{M \times D}$ where D corresponds to the feature dimension.

The segments $\mathcal{S}^{t_1}$ are listed from the point-wise features $\mathcal{F}^{t_1}$. For $\mathcal{S}^{t_1}$, we do not compute mean embeddings but keep the features at point level, using the Transformer encoder to compute point-wise intra-class correspondences. To deal with the attention mechanism large memory requirement, we set a maximum number of P points per segment, which are randomly sampled. After sampling P points for each segment, we input the segments $\mathcal{S}^{t_1}$ point-wise features to the Transformer projection head, followed by another self-attention Transformer encoder as a predictor. We then get for each segment the P point-wise feature

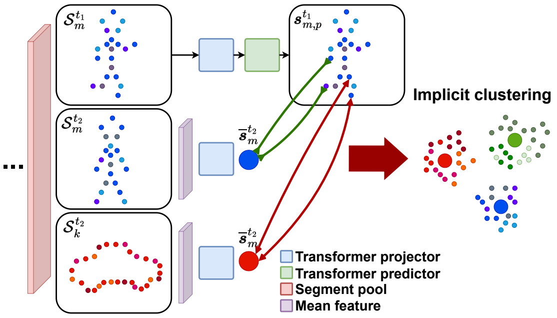

Figure 5. Diagram of the implicit clustering scheme. We pool the segments $\mathcal{S}^{t_1}$ and $\mathcal{S}^{t_2}$ from the backbone features. For t_2, we compute a mean representation for each segment and project it with the Transformer projection head. For t_1, we project the point-wise features and then use the Transformer predictor to predict the corresponding mean representation from t_2, clustering the point-wise features close to its mean target representation.

vectors $\boldsymbol{s}^{t_1} \in \mathbb{R}^{M \times P \times D}$.

With the segment target mean representations $\overline{\boldsymbol{s}}^{t_2}$ and the predicted point-wise feature vectors $\boldsymbol{s}^{t_1}$ for each segment, we compute the loss to minimize the differences between the point features and the corresponding segment mean representation. For each point p from a segment m, we compute the temperature-scaled cosine similarity $\delta_{m,p,k}^{t_1 \to t_2}$ between the normalized point-wise embedding $\boldsymbol{s}_{m,p}^{t_1} \in \mathbb{R}^D$ and the mean representation $\overline{\boldsymbol{s}}_k^{t_2} \in \mathbb{R}^D$ from a segment k as:

$$\delta_{m,p,k}^{t_1 \to t_2} = \frac{(\boldsymbol{s}_{m,p}^{t_1})^\top \overline{\boldsymbol{s}}_k^{t_2}}{\tau}. \tag{1}$$

Next, we use the cross-entropy loss to maximize the similarity between each point p from a segment m and the corresponding target segment mean representation as follows:

$$l_m^{t_1 \to t_2} = -\sum_{p=1}^{P} \log \left(\frac{\exp(\delta_{m,p,m}^{t_1 \to t_2})}{\sum_k^M \exp(\delta_{m,p,k}^{t_1 \to t_2})} \right). \tag{2}$$

We compute this loss for all the M segments and sum it to get the loss for predicting segments from t_2 with the points of the segments from t_1:

$$\mathcal{L}^{t_1 \to t_2} = \sum_{m=1}^{M} l_m^{t_1 \to t_2}. \tag{3}$$

With this formulation, our task is to predict for each point in the segment at time t_1 the corresponding segment at t_2. Since the target for all the P point embeddings $\boldsymbol{s}_{m,p}^{t_1}$ from a segment m is the same mean segment representation $\overline{\boldsymbol{s}}_m^{t_2}$, the loss will push all the points from a segment to converge to a mean representation while separating from other segments, implicitly clustering together points from the same object as shown in Fig. 5.

Lastly, we repeat the forward pass swapping t_1 and t_2 to have a symmetric representation, learning both the correspondences from $t_1 \to t_2$ and from $t_2 \to t_1$ leading to:

$$\mathcal{L}_{TARL} = \mathcal{L}^{t_1 \to t_2} + \mathcal{L}^{t_2 \to t_1}. \tag{4}$$

In the next section we give an intuition behind our choice of projection head, and how it can help the model to learn a more fine-grained representation during pre-training.

3.4. Transformer projection

Self-supervised representation learning methods [13, 14, 27] typically add a non-linear projection head to the backbone to project the embeddings to the target feature space. The idea is that the backbone should learn general features, and this projection head will overfit to the pre-training pretext task and later be discarded during fine-tuning. Our projection scheme follows more recent methods [15,25], which shows that an asymmetric network can improve the learned representation. Besides, with the prior that points from one segment should be similar, we want to focus on the point-wise relationship. Therefore, we replace the non-linear projection head with a self-attention Transformer encoder.

In our siamese network scheme, the online updated network computes point-wise features. In this case, the *queries*, *keys*, and *values* for our Transformer projector will be point-wise features. We aim to learn intra-class features with the attention mechanism and identify correspondences between the points from one segment. In contrast, the momentum updated network computes a mean feature vector for each segment. In this case, the attention mechanism should look for similarities and differences within the different objects segments in the scene. By using the self-attention over points and segments, we aim at learning at the same time point-wise correspondences while also learning the differences between the different segments.

With the Transformer projector, the pre-training can lead the learned features towards a more fine-grained representation, where intra-class point-wise features are matched with the individual segment features. The supplementary material provides ablations comparing the Transformer with the commonly used non-linear projection head.

3.5. Pre-training overview

Secs. 3.1 to 3.4 explain the individual steps of our approach. This section summarizes the entire pre-training pipeline, putting together the modules explained in the previous subsections.

Our pipeline shown in Fig. 2 fetches a batch sample $\mathcal{B}_b$ of n sequential scans and applies a transformation to the scans $\mathcal{P}^t$ with the corresponding poses to have all points in a common coordinate frame. We use an unsupervised ground segmentation approach to remove the ground $\mathcal{G}$. Next, we cluster the remaining points $\hat{\mathcal{P}}$, defining coarse

Method	Scribbles	0.1%	1%	10%	50%	100%
Scratch	54.96	29.35	42.77	53.96	58.27	59.03
PointContrast [67]	54.52	32.63	44.62	58.68	59.98	61.45
DepthContrast [75]	55.90	31.66	48.05	57.11	60.99	61.14
SegContrast [50]	56.70	32.75	44.83	56.31	60.45	61.02
TARL (Ours)	**57.25**	**38.59**	**51.42**	**60.34**	**61.42**	**61.47**

Table 1. Results (mIoU) of the models pre-trained on SemanticKITTI fine-tuned to semantic segmentation on SemanticKITTI with different percentage of labels and scribbles.

Method	Mini	Full
Scratch	26.94	66.03
Supervised pre-training	38.39	67.35
PointContrast [67]	31.92	67.31
DepthContrast [75]	27.81	64.70
SegContrast [50]	31.27	67.70
TARL (Ours)	**39.36**	**68.26**

Table 2. Results (mIoU) of the self-supervised and supervised pre-trained models on SemanticKITTI fine-tuned to semantic segmentation on nuScenes mini and full training sets.

Method	KITTI	nuScenes
Frozen backbone	4.10	5.62
PointContrast [67]	25.53	19.70
DepthContrast [75]	13.87	9.65
SegContrast [50]	27.42	23.75
TARL (Ours)	**35.90**	**26.08**

Table 3. Linear evaluation results (mIoU) on SemanticKITTI and nuScenes datasets for semantic segmentation.

segments $\mathcal{S}$ of the objects in the scene. We sample two random scans at different times $\mathcal{P}^{t_1}$ and $\mathcal{P}^{t_2}$ from $\mathcal{B}_b$, in which we apply random augmentations $\mathcal{T}^{t_1}$ and $\mathcal{T}^{t_2}$. We compute point-wise features $\mathcal{F}^{t_1}$ and $\mathcal{F}^{t_2}$ from the sampled point clouds with the backbone, and list the segments $\mathcal{S}^{t_1}$ and $\mathcal{S}^{t_2}$ with its corresponding point-wise features. We calculate a mean embedding $\overline{\mathcal{S}}^{t_2}$ for each segment m in $\mathcal{P}^{t_2}$ and compute the target representation $\overline{\boldsymbol{s}}^{t_2}$ for each segment with a Transformer projection head. For the segments $\mathcal{S}^{t_1}$, we project the point-wise features with another Transformer projector followed by a Transformer encoder as a predictor to get the point-wise features $\boldsymbol{s}^{t_1}$. Lastly, we compute the loss to predict the corresponding segment representation at t_2 for each point in t_1. We repeat the process changing t_1 and t_2 to symmetrically match points from scan t_2 to the corresponding segment at t_1.

4. Experiments

Datasets. We use the SemanticKITTI dataset [4, 22] for our pre-training, an autonomous driving LiDAR data benchmark with point-wise annotations. During pre-training, we use only the raw point clouds from the training sequences and the given poses to extract the temporal objects views. For fine-tuning, we use SemanticKITTI with the full scans annotations and the scribbles annotations [51]. Besides SemanticKITTI, we also use nuScenes [8, 64] for fine-tuning, to evaluate our approach in different autonomous driving datasets collected with different LiDARs and sensor setups, reporting the results on the validation sets.

Model architecture. For our evaluation, we use a MinkUNet [20] as the backbone, which voxelizes the point clouds and uses sparse convolutions to extract features from the point cloud. The backbone receives the point cloud coordinates and intensity as input, and the output dimension feature has dimension $D = 96$. We use a multi-head self-attention encoder for the projection head with one layer and eight attention heads. For the predictor, we use the same multi-head self-attention encoder architecture. Further details on the backbone used in our experiments are provided in the supplementary material.

Pre-training. For pre-training, we use the AdamW optimizer [43] with a learning rate of $2{\cdot}10^{-4}$ and decay of 10^{-4}, training for 200 epochs with batch size 8 using a single NVIDIA RTX A6000. We use $n = 12$ for the scans to be aggregated. We provide experiments with different numbers of scans in the supplementary material to support this choice. The voxel resolution is set to $0.05\,m$ for the input point clouds, from which we sample a maximum of $40,000$ points per point cloud. During the segment pooling, we limit it to a total of $M = 50$ segments with a maximum of $P = 300$ points per segment to avoid memory overflow. We use $\tau = 0.1$ to compute δ in Eq. (1) and momentum of 0.999 to update the momentum network according to the online network weights. For the baselines, we use their official repositories with their default parameters, also training them for 200 epochs and sampling $40,000$ points per scan.

Fine-tuning. To evaluate our method and compare it with the baselines, we fine-tune the models to three downstream tasks: semantic segmentation, panoptic segmentation, and object detection. For semantic segmentation, we use the pipeline provided in SegContrast [50] with the AdamW optimizer [43] and a learning rate of 0.001. For panoptic segmentation, we use the baseline evaluated by Hong *et al*. [29], where a 3D backbone network is used together with a semantic and an instance heads, followed by a clustering post-processing to identify the instances. We use the pre-trained MinkUNet model as the 3D backbone. We use a learning rate of 0.2 with batch size 8, training for 50 epochs. For object detection, we use the same toolbox and hyperparameters used in DepthContrast [75] with the PartA2 detector [57]. For all the tasks, we evaluate the method with the same model trained from scratch (without pre-training) and with the model after pre-training with

	0.1%		1%		10%		50%		100%	
Method	PQ	IoU	PQ	IoU	PQ	IoU	PQ	IoU	PQ	IoU
Scratch	4.76	11.13	22.72	30.84	47.20	53.53	55.32	61.94	55.40	59.75
PointContrast [67]	5.86	11.51	27.37	32.49	47.57	54.63	54.21	59.48	55.85	61.49
DepthContrast [75]	7.65	13.56	27.31	32.30	46.85	51.27	54.55	59.60	56.15	60.81
SegContrast [50]	7.58	14.46	26.14	32.85	47.02	53.47	55.38	60.04	**56.73**	61.96
TARL (Ours)	**10.26**	**17.01**	**29.24**	**34.71**	**51.27**	**57.59**	**56.10**	**62.36**	56.57	**62.05**

Table 4. Results (PQ and IoU) when fine-tuning the pre-trained models to panoptic segmentation with different percentage of labels on SemanticKITTI.

	Mini		Full	
	PQ	IoU	PQ	IoU
Scratch	23.78	23.96	52.98	58.17
Supervised pre-training	24.77	23.60	53.19	58.05
PointContrast [67]	26.58	25.46	51.06	56.39
DepthContrast [75]	28.66	27.30	51.51	57.06
SegContrast [50]	28.84	26.79	52.31	57.24
TARL (Ours)	**32.22**	**30.73**	**53.26**	**59.14**

Table 5. Results (PQ and IoU) of the self-supervised and supervised pre-trained models on SemanticKITTI fine-tuned to panoptic segmentation on nuScenes mini and full training sets.

the different self-supervised methods. The supplementary material provides detailed information for each downstream task training procedure.

4.1. Semantic segmentation

In this evaluation, we want to measure the information about semantics learned by the network during pre-training. We fine-tune it to semantic segmentation on SemanticKITTI and nuScenes after pre-training on SemanticKITTI. For SemanticKITTI, we use the percentage scans subsets used in SegContrast [50], with 0.1%, 1%, 10%, 50%, and 100% of the labeled scans. We also evaluate on the scribbles annotations [51], which labels just a subset of points for each scan in the SemanticKITTI training sequences. We report the mean intersection-over-union (mIoU) of the models on the validation set, while fine-tuned on the aforementioned subset of labels from the training set.

Tab. 1 shows the results from our approach compared to the baselines. As can be seen, overall the pre-training methods boost the performance compared to the network without pre-training on both scribbles annotations and the subsets of labeled scans. However, our approach surpasses previous state-of-the-art methods in all the different subsets, with a bigger gap when fewer labeled scans are used to fine-tune the model. We can also notice that, after pre-training with our method, only 10% of labeled scans were necessary for the model to surpass the performance of the network trained from scratch with 100% of the labeled scans. This result suggests that our method can effectively reduce the amount of labeled data necessary for the semantic segmentation task by around ten times. In the supplementary material, we also provide the IoU for each individual semantic class.

Generalization. In this experiment, we evaluate the generalization of the learned features. We use the network pre-trained on SemanticKITTI and fine-tune it on nuScenes. We use nuScenes full training set and the mini training subset to fine-tune the network, and evaluate on the full validation set. In this evaluation, we also compare the results of the fully supervised semantic segmentation pre-training on SemanticKITTI fine-tuned on nuScenes. As shown in Tab. 2, even though all methods can improve the performance of the model trained on nuScenes, our approach can surpass previous self-supervised methods by around 8% when training with fewer labels. Besides, compared with supervised pre-training of the network on SemanticKITTI, our approach achieves better performance on both full and mini training sets. These results suggest that our method can replace supervised pre-training for LiDAR data, since it achieved better performance when fine-tuning to a different dataset, collected with a different sensor setup.

Linear evaluation. In this experiment, the pre-trained backbone is frozen, training only a linear head on top of it to evaluate how descriptive the self-supervised learned representation is without fine-tuning it to the target task. We perform this evaluation on SemanticKITTI and nuScenes datasets. In Tab. 3, the randomly initialized network (frozen backbone) achieves low performance, suggesting that the features extracted by the backbone play the main role in achieving semantic segmentation. All the pre-training methods improve the performance compared to the network without pre-training. However, our approach has a clear performance gap compared to the baselines, especially when training to the same dataset used for pre-training. These results suggest that the representation learned by our method can embed more semantic information, even though not using labels, achieving higher performance than previous state-of-the-art methods on both datasets.

4.2. Panoptic segmentation

In this experiment, we fine-tune the pre-trained models for panoptic segmentation to also evaluate the instance-level features of our learned representation. Same as for semantic

	Car			Pedestrian			Cyclist		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Scratch	91.75	82.13	80.16	67.65	63.13	57.73	91.74	74.94	70.34
PointContrast [67]	91.55	81.70	79.88	67.33	62.32	56.55	92.33	75.20	70.66
DepthContrast [75]	91.98	82.04	80.03	**69.13**	63.97	57.71	**92.68**	73.67	70.10
SegContrast [50]	92.08	82.10	**80.18**	68.72	63.77	57.36	91.26	74.70	70.28
TARL (Ours)	**92.09**	**82.15**	80.10	68.50	**64.00**	**58.74**	92.61	**75.44**	**70.97**

Table 6. Results (AP_{50}) when fine-tuning the pre-trained models to object detection on KITTI full training set for car, pedestrian and cyclist classes on the easy, moderate and hard difficult levels.

segmentation, we fine-tuned the model to SemanticKITTI using the same percentage subsets and nuScenes using the full and the mini training sets, reporting both the mIoU and the panoptic quality (PQ) on the validation sets.

Tab. 4 shows that our method is consistently better than previous self-supervised pre-training approaches. As the amount of training samples increases, the differences diminish. However, our method has a clear performance gap compared to the baselines when trained with fewer labels.

Generalization. To evaluate the generalization of our learned features on panoptic segmentation, we use the network pre-trained on SemantiKITTI to fine-tune it on nuScenes full and mini training sets, evaluating it on the full validation set. Tab. 5 shows that the representation learned by our method was more suited to be transferred to a different dataset. As in semantic segmentation, our method achieved better performance than previous self-supervised approaches and supervised pre-training on SemanticKITTI. These results agree with the semantic segmentation evaluation, suggesting that our method is better suited for transfer learning than the supervised pre-training.

With these experiments, we can validate that our learned representations can extract not only semantic knowledge from the data but also instance-level information, surpassing previous methods in IoU and PQ metrics. In the supplementary material, we provide further results, reporting also the segmentation and recognition quality.

4.3. Object detection

As a third downstream task, we compare the methods fine-tuned for object detection to evaluate how general is the representation learned by our method compared with previous state-of-the-art methods. In this evaluation, we fine-tune the model with the whole training set of the KITTI dataset [22] and report the average precision (AP_{50}) with 40 recall positions for car, pedestrian, and cyclist classes on the three difficulty levels.

In Tab. 6, we can compare the baselines performance with our approach. For the car class, the pre-training methods achieve a marginal improvement compared to the network trained from scratch. For pedestrian and cyclist, the improvements brought by the pre-training are more substantial. On those classes, our method achieves the best performance on moderate and hard difficulties, showing that our approach can also boost the performance when fine-tuning the model on object detection. In our supplementary material, we provide further experiments with the different training subsets used by Zhang *et al.* [75].

Together with the evaluations of the other downstream tasks, these results suggest that our method can learn a general representation. Our method achieved better performance than previous approaches on all three downstream tasks, showing that our approach learns a robust representation, not biased to a specific task but showing generalizable representations suitable for different tasks and datasets.

5. Conclusion

In this paper, we propose a self-supervised representation learning method for 3D LiDAR data evaluated in the context of autonomous driving. We exploit the vehicle motion to extract different views of objects across time to learn a temporally-consistent representation. We use a Transformer encoder as a projection head and implicitly clustered points from the same object together while discriminating between different objects. Our experiments show that our method achieves better performance than previous state-of-the-art approaches on different downstream tasks. Besides, our approach could reduce the amount of necessary labeled data to only 10% when fine-tuning for semantic segmentation on SemanticKITTI. These results show that our proposed approach could learn a general LiDAR point cloud representation, embedding semantic information by matching objects representation viewed at different times. In addition, our method achieved better performance than supervised pre-training when transferring the learned representation to a dataset collected with different LiDAR sensors, suggesting that our approach could replace supervised pre-training in the LiDAR data domain.

Limitations. Despite encouraging results, there is space for further improvement. For the unsupervised segment extraction, the clustering parameters have to be tuned depending on the LiDAR used to collect the data. Also, we rely on high-frequency data to properly segment dynamic objects. Otherwise, temporal segments could be wrongly matched.

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2021. 2

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2017. 2

[3] Mehmet Aygun, Aljosa Osep, Mark Weber, Maxim Maximov, Cyrill Stachniss, Jens Behley, and Laura Leal-Taixe. 4D Panoptic LiDAR Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[4] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Juergen Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019. 1, 2, 6

[5] Jens Behley and Cyrill Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2018. 3, 4

[6] Benedikt Mersch, Xieyuanli Chen, Ignacio Vizzo, Lucas Nunes, Jens Behley, and Cyrill Stachniss. Receding Moving Object Segmentation in 3D LiDAR Data Using Sparse 4D Convolutions. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7503–7510, 2022. 2

[7] Borna Bešić, Nikhil Gosala, Daniele Cattaneo, and Abhinav Valada. Unsupervised Domain Adaptation for LiDAR Panoptic Segmentation. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022. 3

[8] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 6

[9] Ricardo J. G. B. Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, 2013. 3

[10] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 1, 2, 4

[11] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 1, 2

[12] Changqing Zhou, Zhipeng Luo, Yueru Luo, Tianrui Liu, Liang Pan, Zhongang Cai, Haiyu Zhao, and Shijian Lu. PTTR: Relational 3D Point Cloud Object Tracking With Transformer. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2020. 1, 2, 5

[14] Xinlei Chen, Haoqi Fan, Ross B. Girshick, and Kaiming He. Improved Baselines with Momentum Contrastive Learning. *arXiv preprint*, arxiv:2003.04297, 2020. 1, 2, 5

[15] Xinlei Chen and Kaiming He. Exploring Simple Siamese Representation Learning. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2, 5

[16] Xieyuanli Chen, Shijie Li, Benedikt Mersch, Louis Wiesmann, Jürgen Gall, Jens Behley, and Cyrill Stachniss. Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data. *IEEE Robotics and Automation Letters (RA-L)*, 6:6529–6536, 2021. 2

[17] Xieyuanli Chen, Andres Milioto, Emanuele Palazzolo, Philippe Giguère, Jens Behley, and Cyrill Stachniss. SuMa++: Efficient LiDAR-based Semantic SLAM. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019. 3, 4

[18] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked Feature Prediction for Self-Supervised Visual Pre-Training. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[19] Cheng Zhang, Haocheng Wan, Xinyi Shen, and Zizhao Wu. PatchFormer: An Efficient Point Transformer With Patch Attention. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[20] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 6

[21] Fayao Liu, Guosheng Lin, and Chuan-Sheng Foo. Point discriminative learning for unsupervised representation learning on 3d point clouds. *arXiv preprint*, arXiv:2108.02104, 2021. 1, 2

[22] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012. 1, 2, 6, 8

[23] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[24] Benjamin Graham and Laurens van der Maaten. Submanifold Sparse Convolutional Networks. *arXiv preprint*, arxiv:1706.01307, 2017. 2

[25] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and

Michal Valko. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 1, 2, 5

[26] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J. Kusner. Unsupervised Point Cloud Pre-Training via Occlusion Completion. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 2

[27] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 5

[28] Hiroharu Kato and Tatsuya Harada. Image Reconstruction from Bag-of-Visual-Words. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[29] Fangzhou Hong, Hui Zhou, Xinge Zhu, Hongsheng Li, and Ziwei Liu. LiDAR-Based Panoptic Segmentation via Dynamic Shifting Network. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 6

[30] Hongxu Yin, Arash Vahdat, Jose Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-ViT: Adaptive Tokens for Efficient Vision Transformer. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[31] Ji Hou, Benjamin Graham, Matthias Niessner, and Saining Xie. Exploring Data-Efficient 3D Scene Understanding With Contrastive Scene Contexts. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2, 3

[32] Ignacio Vizzo, Tiziano Guadagnino, Benedikt Mersch, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *arXiv preprint arXiv:2209.15397*, 2022. 3, 4

[33] Ishan Misra and Laurens van der Maaten. Self-Supervised Learning of Pretext-Invariant Representations. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[34] Ishan Misra, Rohit Girdharm, and Armand Joulin. An End-to-End Transformer Model for 3D Object Detection. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 2

[35] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. ST3D: Self-Training for Unsupervised Domain Adaptation on 3D Object Detection. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[36] Junbo Yin, Dingfu Zhou, Liangjun Zhang, Jin Fang, Cheng-Zhong Xu, Jianbing Shen, and Wenguan Wang. Proposalcontrast: Unsupervised pre-training for lidar-based 3d object detection. *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022. 1, 2, 3

[37] Kai Fischer, Martin Simon, Florian Oelsner, Stefan Milz, Horst-Michael Gross, and Patrick Maeder. StickyPillars: Robust and Efficient Feature Matching on Point Clouds Using Graph Neural Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[38] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast Encoders for Object Detection From Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[39] Ferdinand Langer, Andres Milioto, Alexandre Haag, Jens Behley, and Cyrill Stachniss. Domain Transfer for Semantic Segmentation of LiDAR Data using Deep Neural Networks. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020. 3

[40] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph Attention Convolution for Point Cloud Semantic Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[41] Hyungtae Lim, Sungwon Hwang, and Hyun Myung. ERASOR: Egocentric Ratio of Pseudo Occupancy-Based Dynamic Object Removal for Static 3D Point Cloud Map Building. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):2272–2279, 2021. 3

[42] Ling Zhang and Zhigang Zhu. Unsupervised feature learning for point cloud understanding by contrasting and clustering using graph convolutional neural networks. In *Proc. of the Intl. Conf. on 3D Vision (3DV)*, 2019. 1, 2

[43] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2019. 6

[44] Rodrigo Marcuzzi, Lucas Nunes, Louis Wiesmann, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Contrastive Instance Association for 4D Panoptic Segmentation using Sequences of 3D LiDAR Scans. *IEEE Robotics and Automation Letters (RA-L)*, 7(2):1550–1557, 2022. 2

[45] Andres Milioto, Jens Behley, Chris McCool, and Cyrill Stachniss. LiDAR Panoptic Segmentation for Autonomous Driving. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020. 1

[46] Andres Milioto, Leonard Mandtler, and Cyrill Stachniss. Fast Instance and Semantic Segmentation Exploiting Local Connectivity, Metric Learning, and One-Shot Detection for Robotics. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019. 1, 2

[47] Andres Milioto and Cyrill Stachniss. Bonnet: An Open-Source Training and Deployment Framework for Semantic Segmentation in Robotics using CNNs. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019. 2

[48] Mingchen Zhuge, Dehong Gao, Deng-Ping Fan, Linbo Jin, Ben Chen, Haoming Zhou, Minghui Qiu, and Ling Shao. Kaleido-BERT: Vision-Language Pre-Training on Fashion Domain. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[49] Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. CrossPoint: Self-Supervised Cross-Modal Contrastive Learning for 3D Point Cloud Understanding. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[50] Lucas Nunes, Rodrigo Marcuzzi, Xieyuanli Chen, Jens Behley, and Cyrill Stachniss. SegContrast: 3D Point Cloud Feature Representation Learning through Self-supervised Segment Discrimination. *IEEE Robotics and Automation Letters (RA-L)*, 2022. 1, 2, 3, 6, 7, 8

[51] Ozan Unal, Dengxin Dai, and Luc Van Gool. Scribble-supervised lidar semantic segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 6, 7

[52] Pedro O. Pinheiro, Amjad Almahairi, Ryan Y. Benmalek, Florian Golemo, and Aaron Courville. Unsupervised learning of dense visual representations. In *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, pages 4489–4500, 2020. 1, 2

[53] Pierre Dellenbach, Jean-Emmanuel Deschaud, Bastien Jacquet, and François Goulette. CT-ICP Real-Time Elastic LiDAR Odometry with Loop Closure. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022. 3, 4

[54] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

[55] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2017. 1, 2

[56] Aditya Sanghi. Info3d: Representation learning on 3d objects using mutual information maximization and contrastive learning. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pages 626–642, 2020. 1, 2

[57] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From Points to Parts: 3D Object Detection from Point Cloud with Part-aware and Part-aggregation Network. *IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI)*, 43:2647–2664, 2021. 2, 6

[58] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[59] Sukmin Yun, Hankook Lee, Jaehyung Kim, and Jinwoo Shin. Patch-Level Representation Learning for Self-Supervised Vision Transformers. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[60] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2020. 1, 2

[61] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019. 1, 2

[62] Weijing Shi and Ragunathan (Raj) Rajkumar. Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[63] Wenguan Wang, Tianfei Zhou, Fisher Yu, Jifeng Dai, Ender Konukoglu, and Luc Van Gool. Exploring Cross-Image Pixel Contrast for Semantic Segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 1, 2

[64] Whye Kit Fong, Rohit Mohan, Juana Valeria Hurtado, Lubing Zhou, Holger Caesar, Oscar Beijbom, Abhinav Valada. Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *IEEE Robotics and Automation Letters (RA-L)*, 7(2):3795–3802, 2022. 2, 6

[65] Xiangyun Zhao, Raviteja Vemulapalli, Philip Mansfield, Boqing Gong, Bradley Green, Lior Shapira, and Ying Wu. Contrastive Learning for Label Efficient Semantic Segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 1, 2

[66] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling Vision Transformers. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[67] Saining Xie, Jiatao Gu, Demi Guo, Charles R. Qi, Leonidas Guibas, and Or Litany. PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2020. 1, 2, 3, 6, 7, 8

[68] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 2

[69] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense Contrastive Learning for Self-Supervised Visual Pre-Training. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2

[70] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-BERT: Pre-Training 3D Point Cloud Transformers With Masked Point Modeling. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[71] Li Yi, Boqing Gong, and Thomas Funkhouser. Complete & Label: A Domain Adaptation Approach to Semantic Segmentation of LiDAR Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[72] Youngwan Lee, Jonghee Kim, Jeffrey Willette, and Sung Ju Hwang. MPViT: Multi-Path Vision Transformer for Dense Prediction. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[73] Xiaoding Yuan, Adam Kortylewski, Yihong Sun, and Alan Yuille. Robust Instance Segmentation Through Reasoning About Multi-Object Occlusion. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[74] Yujin Chen, Matthias Nießner, and Angela Dai. 4dcontrast: Contrastive learning with dynamic correspondences for 3d scene understanding. *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022. 1, 2, 3

[75] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. Self-Supervised Pretraining of 3D Features on any

Point-Cloud. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 1, 2, 3, 6, 7, 8

[76] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2021. 1, 2

[77] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 2

[78] Zhaoqing Wang, Qiang Li, Guoxin Zhang, Pengfei Wan, Wen Zheng, Nannan Wang, Mingming Gong, and Tongliang Liu. Exploring Set Similarity for Dense Self-Supervised Representation Learning. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[79] Zixiang Zhou, Yang Zhang, and Hassan Foroosh. Panoptic-PolarNet: Proposal-Free LiDAR Point Cloud Panoptic Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[80] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and Asymmetrical 3D Convolution Networks for LiDAR Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2