# Self-Supervised Image-to-Point Distillation via Semantically Tolerant Contrastive Loss

Anas Mahmoud[1], Jordan S. K. Hu[1], Tianshu Kuai[1], Ali Harakeh[2], Liam Paull[2], and Steven L. Waslander[1]
[1]University of Toronto Robotics Institute, [2]Mila, Université de Montréal

## Abstract

*An effective framework for learning 3D representations for perception tasks is distilling rich self-supervised image features via contrastive learning. However, image-to-point representation learning for autonomous driving datasets faces two main challenges: 1) the abundance of self-similarity, which results in the contrastive losses pushing away semantically similar point and image regions and thus disturbing the local semantic structure of the learned representations, and 2) severe class imbalance as pretraining gets dominated by over-represented classes. We propose to alleviate the self-similarity problem through a novel semantically tolerant image-to-point contrastive loss that takes into consideration the semantic distance between positive and negative image regions to minimize contrasting semantically similar point and image regions. Additionally, we address class imbalance by designing a class-agnostic balanced loss that approximates the degree of class imbalance through an aggregate sample-to-samples semantic similarity measure. We demonstrate that our semantically-tolerant contrastive loss with class balancing improves state-of-the-art 2D-to-3D representation learning in all evaluation settings on 3D semantic segmentation. Our method consistently outperforms state-of-the-art 2D-to-3D representation learning frameworks across a wide range of 2D self-supervised pretrained models.*

## 1. Introduction

Self-supervised learning (SSL) has shown significant success in learning useful representations from unlabeled images [7, 9, 11, 22], mainly due to large, diverse, and balanced 2D image datasets. These successes promise to alleviate the requirement for large labeled datasets, which can be expensive, not attainable, or task-specific. These issues are exacerbated when generating labels for 3D point clouds, which are usually much more difficult to annotate [27] than 2D images. Additionally, the sparse nature of point clouds generated using a LiDAR sensor, as is common in outdoor autonomous driving data, substantially increases the difficulty of manually generating per-point labels, particularly at large distances.

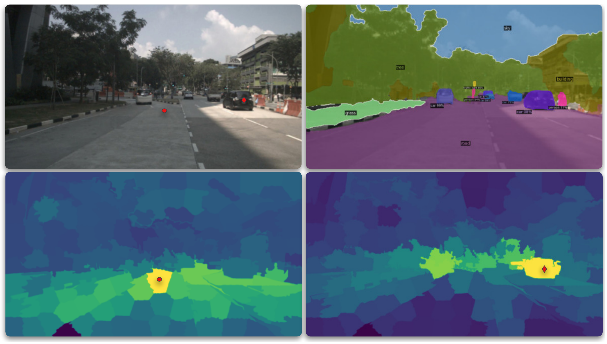

Figure 1. **Bottom row:** Superpixel-to-superpixel cosine similarity with respect to, **bottom left:** a road anchor, and **bottom right:** a vehicle anchor (both marked in red). Superpixel-driven contrastive loss [21] treats all superpixels excluding the anchor as negative samples. As such, loss will be dominated by gradients from semantically similar negative samples, disturbing the local semantic structure of the learned 3D representations. Our loss uses superpixel similarity to 1) Reduce the contribution of false negative samples, and 2) Balance the contribution of well-represented (i.e., road) and under-represented (i.e., vehicle) anchors

A common approach to learn 3D representations is through multimodal invariance [16], where 3D representations are learned to be invariant to features extracted from image encoders trained with self-supervised learning [17, 21]. The current state-of-the-art, SLidR [21], encourages learning representations of 3D point regions (superpoints) to match pre-trained representations of 2D image regions (superpixels) through a novel contrastive loss. By contrasting 2D and 3D regions, SLidR [21] enables learning representations from point clouds with varying point densities, as is common in autonomous driving datasets.

Unfortunately, SlidR's region-based sampling does not address self-similarity, which results when fewer unique semantic classes exist in the data relative to the number of chosen contrastive pairs during training. Under self-similarity, many negative samples will belong to the same semantic class as the positive sample used to compute the contrastive loss, pushing apart their embeddings and breaking the local semantic structure of learned 3D representation [24](see Figure 1). This issue is further exacerbated by the implicit hardness-aware property of contrastive loss, where the largest gradient contributions come from the most semantically similar negative samples [24](see 3.1.2).

In addition, autonomous driving datasets are highly imbalanced, for example, in the nuScenes dataset [6], the 'Pedestrian' class covers 0.25% of the data, while 'vegetation' class covers 22.19% of the data. Since the class of the positive sample is unknown during pretraining, SLidR's loss gives an equal weight to all samples in the batch. Hence, the 3D pretraining is predominately driven by gradients from a few over-represented samples, leading to poor performance on under-represented samples.

In this work, we simultaneously address the challenge of contrasting semantically similar point and image regions and the challenge of learning 3D representations from highly imbalanced autonomous driving datasets. Figure 1 shows that image regions semantically similar to the anchor exhibit high cosine similarity in the 2D self-supervised feature space. Our first key idea is to exploit the semantic distance between positive and negative pooled image features to guide negative sample selection. Reducing the contribution of false negative samples, which are abundant in autonomous driving datasets due to the self-similarity, prevents the disturbance of the local semantic structure of the pre-trained 3D representations [24]. Figure 2 shows that most anchors come from over-represented classes (i.e., road, vegetation) resulting in a 3D point encoder that is less discriminative with respect to under-represented classes. To address this challenge, we propose using aggregate semantic similarity between samples as a proxy for class imbalance. By balancing the contribution of over and under-representated anchors, we improve the learned 3D representations of under-represented semantic classes (i.e., pedestrians and vehicles). We summarize our approach with two main contributions, which we present below.

**Semantically-Tolerant Loss**. To address the similarity of samples in 2D-to-3D representation learning frameworks, we propose a novel contrastive loss that relies on 2D self-supervised image features to infer the semantic distance between positive and negative pooled image features. We propose to either directly reduce the gradient contribution of semantically-similar negative samples or exclude the K-nearest samples based on the semantic distance to the positive sample.

**Class Agnostic Balanced Loss**. To address pre-training using highly imbalanced 3D data, we propose a novel class agnostic balancing for contrastive losses that weights the contribution of each 3D region in a point cloud based on the aggregate semantic similarity of its corresponding 2D region with all negative samples. We reason that samples with high aggregate semantic similarity to other samples come from over-represented classes, while under-represented samples are semantically similar to very few other samples. Hence, we reduce the contribution of over-represented samples, while increasing that of under-represented samples.

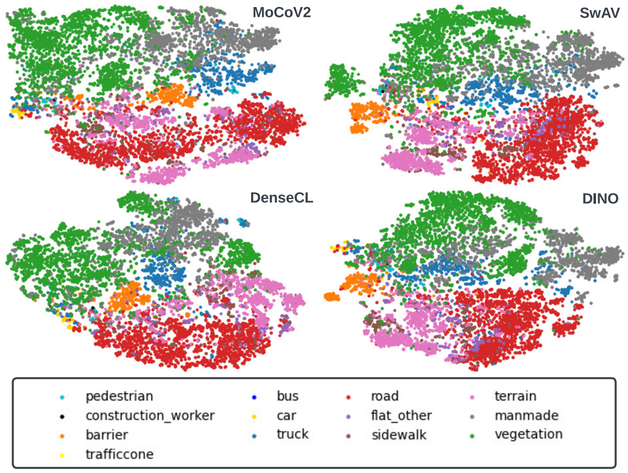

Figure 2. t-SNE [23] visualization of superpixel-level features for a given batch of nuScenes [6] images. Each superpixel feature is colorized based on its semantic class derived from LiDAR ground truth point-wise labels. Here, we show that MoCoV2 [10], SwAV [7], DenseCL [25] and DINO [8] weights generate meaningful semantic clusters on the superpixel level.

By extending the state-of-the-art 2D-to-3D representation learning frameworks using our proposed semantically-tolerant contrastive loss with class balancing, we show that we can improve their performance on in-distribution linear probing and finetuning semantic segmentation, as well as on out-of-distribution few-shot semantic segmentation. We also show that our proposed semantically-tolerant loss improves 3D semantic segmentation performance across a wide range of 2D self-supervised pretrained image features, consistently outperforming state-of-the-art 2D-to-3D representation learning frameworks.

## 2. Related Work

### 2.1. Self-Supervised 2D Representation Learning

Learning 2D representations via instance-level discrimination has shown to be an effective pre-text task for SSL frameworks [26]. These frameworks learn unsupervised representations by maximizing the mutual information [3] between two augmented views of an image using one of two objectives. Similarity maximization objective including contrastive methods [9, 10, 25] minimize the distance between the representations of the two views of the same instance, while maximizing the distance to other instances. On the other hand, redundancy minimization [4, 28] objective minimizes the statistical correlation between the dimensions of the learned representation while also maximizing similarity between representations of the same instance. SSL frameworks can also be categorized based on whether their losses are designed to discriminate pixel-level [25], superpixel-level [13] or instance-level [9] representations. DetCon [13] has demonstrated that designing contrastive losses on image regions named superpixels leads to efficient pretraining and better performance especially on dense tasks like 2D semantic segmentation and object detection compared to instance-level contrastive losses.

In this paper, we employ the self-supervised 2D representations as a supervisory signal to learn 3D representations. We demonstrate that the semantic structure learned by 2D SSL methods can be used to address challenges in learning 3D representations for autonomous driving datasets including abundance of self-similarity and severe class imbalance.

### 2.2. Self-Supervised 3D Representation Learning

Self-supervised 3D representation learning can be categorized into perspective-invariant, format-invariant and multimodal invariant methods [16]. Perspective invariant methods like PointContrast [27] learn point representations that are invariant to different views of the point cloud. These methods are primarily designed for indoor RGB-D datasets, where full 3D reconstruction of the scene is possible [29]. To address the limitation of requiring multiple views, DepthContrast [29], a format-invariant method, uses only single view point cloud data, and learns 3D representations that are invariant to point and voxel representations. By design, the contrastive loss in DepthContrast [29] learns global scene-level representations and therefore is prone to losing information on small objects [21].

Multi-modal invariant methods extract image representations from pretrained image encoders, and use a contrastive loss to learn 3D representations by maximizing the similarity to 2D representations. PPKT [17] contrasts representations of pairs of point and pixel correspondences. This method is mainly designed for indoor RGB-D datasets, where dense point-to-pixel correspondences exist [21]. SLidR [21] is the first 3D representation learning method that is primarily designed for autonomous driving datasets. Inspired by DetCon [13], they use unsupervised image segmentation algorithms [1] to segment images into superpixels. By projecting the point cloud onto the superpixel mask, each point is assigned a superpoint. Each superpoint and its corresponding superpixel form a positive pair and the contrastive loss is thus defined at the superpixel-level. SLidR [21] formulation has multiple advantages; First, constructing semantically coherent image and point cloud corresponding regions, leads to learning useful object-level representations [13]. In addition, unlike PPKT [17], grouping superpoint and superpixel representations using average pooling increases robustness against camera and LiDAR calibration errors [21]. Finally, the density of LiDAR returns increases as a function of distance [14, 18, 19] resulting in very few number of points at mid-to-long range objects. The Random sampling strategy of positive pairs in PointContrast [27] and PPKT [17] applied to outdoor scenes results in a biased sampling towards dense nearby points. SLidR [21] breaks down the scene based on image superpixels, which leads to constrastive pairs covering the entire 3D scene. Each pair has the same weight in the contrastive loss regardless of the number of points in these regions [21].

The contrastive losses proposed in PPKT [17] and SLidR [21] are not suited for autonoumous driving data due to two main reasons. First, the high level of self-similarity, which results in the number contrastive pairs from unique semantic classes being much less than the number of pairs in any given batch. This phenomenon will lead to the contastive loss considering semantically similar samples as negative samples and pushing their representations apart. Second, autonomous driving datasets suffer from severe class imbalance which can lead to a small over-represented subset of the semantic class dominating the self-supervised pretraining stage. We address the first challenge by formulating a semantically tolerant loss which prevents contrasting semantically similar samples. We address the second challenge by formulating a class-agnostic balanced loss that uses aggregate sample-to-samples semantic distance as a proxy for class imbalance.

## 3. Methodology

### 3.1. Superpixel-Driven Contrastive Loss

#### 3.1.1 Background

Given a set of point clouds representing multiple scenes $\{\mathbf{p}_i = \{\boldsymbol{\ell}_i, \mathbf{f}_i\} \mid i = 1, \ldots, U\}$, where $\boldsymbol{\ell}_i \in \mathbb{R}^{N_i \times 3}$ is a tensor of 3D location of $N_i$ points representing the $i^{th}$ scene, and $\mathbf{f}_i \in \mathbb{R}^{N_i \times L}$ are their associated point-wise features (i.e., intensity and elongation). We also have a set of camera-to-LiDAR synchronized images $\{\{\mathbf{I}_i^1, \ldots, \mathbf{I}_i^J\} \mid i = 1, \ldots, U\}$, where $\mathbf{I}_i^j \in \mathbb{R}^{H \times W \times 3}$ and $J$ is the number of cameras per scene. Using the unsupervised segmentation algorithm SLIC [1], each pixel of image $\mathbf{I}_i^j$ is segmented into a set of superpixels $\mathcal{X}_i^j$ and $\{\mathcal{X}_i = \{\mathcal{X}_i^1, \ldots, \mathcal{X}_i^J\} \mid i = 1, \ldots, U\}$ denotes the set of multi-scene superpixels $\mathcal{X}$. To generate the set of corresponding superpoints $\{\mathcal{P}_i = \{\mathcal{P}_i^1, \ldots, \mathcal{P}_i^J\} \mid i = 1, \ldots, U\}$, camera-to-LiDAR calibration matrices are used to map 3D points to pixel locations. Superpixels with no corresponding superpoints (i.e., superpixels outside FOV of the LiDAR) are removed from set $\mathcal{X}$ and therefore $|\mathcal{X}| = |\mathcal{P}|$.

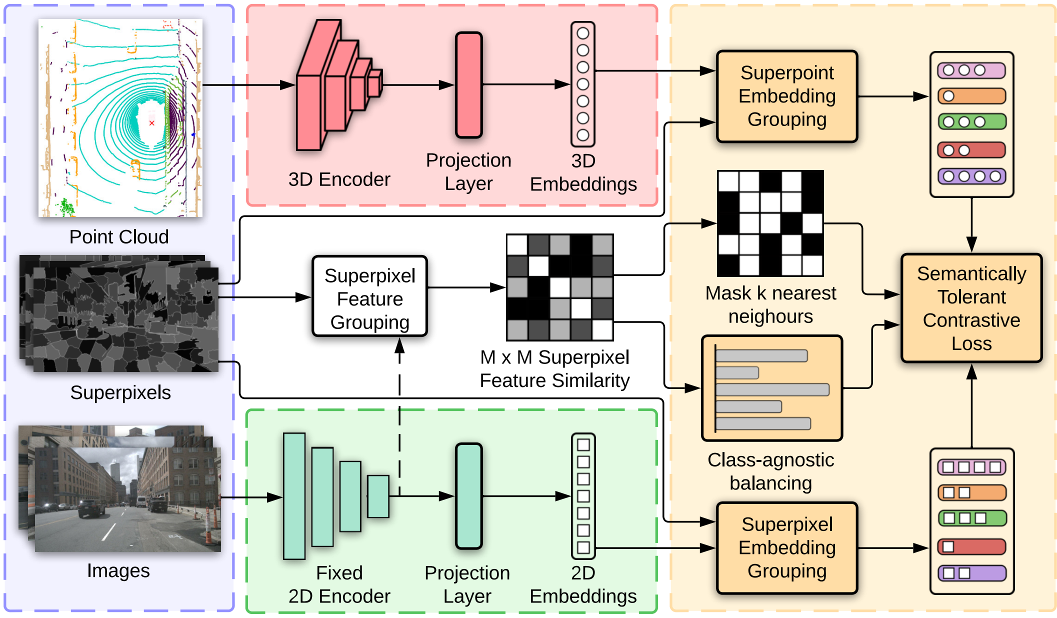
Point Cloud
Superpixels
Images
3D Encoder
Projection Layer
3D Embeddings
Superpixel Feature Grouping
M x M Superpixel Feature Similarity
Fixed 2D Encoder
Projection Layer
2D Embeddings
Superpoint Embedding Grouping
Mask k nearest neighbours
Class-agnostic balancing
Semantically Tolerant Contrastive Loss
Superpixel Embedding Grouping

Figure 3. An overview of our self-supervised image-to-point distillation framework. LiDAR and camera data are encoded and their respective features are projected onto an embedding space where the contrastive loss is computed. Then, superpixels are used to group the 3D and 2D embeddings. In addition, the fixed 2D self-supervised features are grouped and used to estimate the superpixel-to-superpixel similarity. Finally, our loss utilizes the similarity estimates to: 1) Reduce the contribution of false negative superpixel embeddings by masking the k-nearest neigbouring superpixels, and 2) Balance the contribution of over and under-represented samples based on the distribution of the aggregate superpixel feature similarities.

Let the point cloud encoder be a 3D deep neural network $f_{\theta_P} : \mathbb{R}^{N\times(3+L)} \rightarrow \mathbb{R}^{N\times D}$, with randomly initialized, trainable parameters $\theta_P$. Let the image encoder be a 2D neural network $g_{\theta_I} : \mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{\frac{H}{s}\times\frac{W}{s}\times C}$, with parameters $\theta_I$ initialized from any 2D self-supervised pretrained parameters. The goal is to use the pretrained image features at the superpixel level as a supervisory training signal for the point cloud encoder.

To compute the superpixel-driven contrastive loss, a trainable projection layer $h_{\omega_P} : \mathbb{R}^{N\times D} \rightarrow \mathbb{R}^{N\times E}$ maps the output of the point cloud encoder to the contrastive loss embedding space. In addition, a trainable projection layer $h_{\omega_I} : \mathbb{R}^{\frac{H}{s}\times\frac{W}{s}\times C} \rightarrow \mathbb{R}^{\frac{H}{s}\times\frac{W}{s}\times E}$ maps pixel-level image features to the contrastive loss embedding space. The loss is computed between superpoint and superpixel embeddings. First, $\mathcal{P}$ and $\mathcal{X}$ are used to group point and pixel embeddings respectively. Let $M = |\mathcal{X}| = |\mathcal{P}|$. An average pooling function is then applied to the point and pixel embeddings within each group, to extract multi-scene superpoint embeddings $\mathbf{Q} \in \mathbb{R}^{M\times E}$ and superpixel embeddings $\mathbf{K} \in \mathbb{R}^{M\times E}$. The superpixel-driven loss is formulated as:

$$\mathcal{L}\left(\mathbf{Q},\mathbf{K}\right) = -\frac{1}{M}\sum_{i=0}^{M}\log\left[\frac{e^{(\langle\mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}{\sum_{j\neq i} e^{(\langle\mathbf{q}_i,\mathbf{k}_j\rangle/\tau)} + e^{(\langle\mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}\right], \quad (1)$$

where $\langle\mathbf{q}_i,\mathbf{k}_j\rangle$ is a measure of similarity computed as the dot product between the $\ell_2$-normalized superpoint and superpixel embeddings and $\tau$ is the temperature scale [9].

#### 3.1.2 Limitations

**Self-similarity in Autonomous Driving Data.** Looking at Figure 1, we observe that many image regions defined by superpixels belong to the same semantic class. We call this self-similarity. For a given batch, each superpoint and its corresponding superpixel embedding are considered positive samples, while the remaining pairs are treated as negative samples. Due to the self-similarity, there is a high probability for the contrastive loss in SLidR [21] (and a higher probability in PPKT [17]) of pushing away semantically similar samples.

**Hardness-aware Property of Contrastive Loss.** The success of the softmax-based contrastive loss has been attributed to its hardness-aware property [24]. The temperature parameter $\tau$ controls the distribution of negative gradients, where low temperatures lead to larger gradient contribution from nearest neighbour negative samples. Authors in [24] have demonstrated that there exists a uniformity-tolerance dilemma in softmax-based contrastive losses. They show that high temperatures lead to semantically tolerant embeddings, but can suffer from embedding collapse, while low temperatures lead to a uniform distribution of embeddings, preventing embedding collapse. Nonetheless, low temperatures also lead to less tolerant embeddings, where similar samples are not closely clustered. Both PPKT [17] and SLidR [21] use a low temperature of $\tau = 0.04$ and $\tau = 0.07$ respectively. Due to the implicit hardness-aware property of the contrastive loss, the highest gradient contribution to the pre-training signal comes from pushing away semantically similar samples which disturbs the local semantic structure of the learned representation.

### 3.2. Semantically-Tolerant Contrastive Loss

We observe that in self-supervised image-to-point cloud knowledge distillation frameworks [17, 21], pre-trained models provide a strong prior on the relationship between superpixel features. To show this, we use ResNet-50 pre-trained on ImageNet [20] using four 2D self-supervised methods. First, we use each pre-trained model to map a batch of 16 images from nuScenes dataset [6] to output features $g_{\theta_I} : \mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{\frac{H}{s}\times\frac{W}{s}\times C}$. Using $\mathcal{X}$, pixel features are then grouped and an average pooling function is applied to extract superpixel features $\mathbf{F} \in \mathbb{R}^{M\times C}$. To visualize $\mathbf{F}$, we reduce the dimensionality of the features using t-SNE [23] and show the first 2 dimensions in Figure 2. Each point corresponds to a superpixel feature $\mathbf{f}_i \in M \times C$ colorized using the dominant point-wise ground truth label of their corresponding superpoint.

As seen in Figure 2, extracted superpixel features from nuScenes [6] images form relatively coherent semantic clusters. SLidR [21] not only ignores this strong prior in their contrastive loss formulation but also suffers due to the hardness-aware property [24] of the contrastive loss, which will in its current form, primarily focus on pushing away semantically similar superpoints and superpixels embeddings. In addition, we observe a high level of self-similarity, where the ratio between the total number of superpixels (i.e., 9000) and the number of unique semantic classes (i.e., 13) in a batch is very high, leading to an increase in false negatives.

**Similarity-aware Loss**. Our goal is to address the issue of contrasting semantically similar superpoint and superpixel embeddings. This issue is exacerbated due to the high self-similarity in autonomous driving data, and the hardness-aware property of the contrastive loss. We propose a semantically-tolerant contrastive loss that utilizes superpixel similarities in the feature space to re-weight the contribution of semantically similar negative samples. Our loss reduces the gradient contribution from negative superpixel embeddings that are semantically similar to the positive sample. Our loss can be written as:

$$\mathcal{L}_\alpha\left(\mathbf{Q},\mathbf{K}\right) = \\ -\frac{1}{M}\sum_{i=0}^{M}\log\left[\frac{e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}{\sum_{j\neq i} e^{((1-\alpha_{ij})\,\cdot\,\langle \mathbf{q}_i,\mathbf{k}_j\rangle/\tau)} + e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}\right], \tag{2}$$

where $\alpha_{ij} = \langle \mathbf{f}_i, \mathbf{f}_j\rangle$ is a measure of similarity defined as the dot product of the $\ell_2$ normalized superpixel features $\mathbf{F}$. Here, $\mathbf{f}_i \in \mathbb{R}^C$.

$$P_{ij} = \left[\frac{e^{((1-\alpha_{ij})\,\cdot\,\langle \mathbf{q}_i,\mathbf{k}_j\rangle/\tau)}}{\sum_{j\neq i} e^{((1-\alpha_{ij})\,\cdot\,\langle \mathbf{q}_i,\mathbf{k}_j\rangle/\tau)} + e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}\right]. \tag{3}$$

Looking at the gradient of the proposed loss with respect to the negative similarity $\langle \mathbf{q}_i, \mathbf{k}_j\rangle$ where $j \neq i$:

$$\frac{\partial \mathcal{L}_\alpha\left(\mathbf{q}_i,\mathbf{K}\right)}{\partial\left\langle \mathbf{q}_i,\mathbf{k}_j\right\rangle} = \frac{(1-\alpha_{ij})}{\tau}P_{ij}. \tag{4}$$

In $\mathcal{L}_\alpha$, the gradient of the loss with respect to the negative sample $\mathbf{k}_j$, is weighted by the dissimilarity between the features of positive superpixel $\mathbf{f}_i$ and the negative superpixel $\mathbf{f}_j$. Therefore, there is less contribution from $\mathbf{k}_j$'s that are semantically similar to $\mathbf{k}_i$. When $\alpha_{ij} = 1.0$ the pos/neg superpixel features are identical, and $\frac{\partial \mathcal{L}_\alpha(\mathbf{q}_i,\mathbf{K})}{\partial\langle \mathbf{q}_i,\mathbf{k}_j\rangle} = 0.0$ preventing contrasting semantically similar pairs. When $\alpha_{ij} = 0.0$ then $\frac{\partial \mathcal{L}_\alpha(\mathbf{q}_i,\mathbf{K})}{\partial\langle \mathbf{q}_i,\mathbf{k}_j\rangle} = \frac{P_{ij}}{\tau}$ and we revert back to the SLidR [21] formulation.

**Nearest-Neighbour-aware Loss**. Training with $\mathcal{L}_\alpha$ is a much easier loss to minimize compared to $\mathcal{L}$ since the closest negative samples which are also the hardest negatives have lower contribution to the loss. For instance, when the mean of superpixel-to-superpixel feature similarities is high for a given 2D SSL pretrained model, $\sum_{j\neq i} e^{((1-\alpha_{ij})\,\cdot\,\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)} << e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}$ and thus very few negative samples will contribute to the loss. In this case, $\mathcal{L}_\alpha$ can be easily minimized without learning useful representations. One approach to address this issue, is to suppress negative samples with low similarity values by setting $\alpha_{ij} < \alpha_{min}$ to 0.0, where $\alpha_{min}$ is a tunable parameter. Hence, we ensure we have enough negative samples to learn useful representations while preventing contrasting against semantically similar negative samples. However, $\alpha_{min}$ has to be tuned for each 2D pretrained model. Empirically, we find that the linear separability of the 3D representations is very sensitive to the choice of $\alpha_{min}$ (see Table 6).

We hypothesize that the values of superpixel-to-superpixel similarities are not well-calibrated and vary based on the 2D pretrained model. Therefore, these similarities should not be directly used in the contrastive loss. For a given positive sample, we hypothesize that the order of similarities across different 2D pretrained models is consistent. Hence, the order of similarities is more informative than the similarity values. To avoid directly incorporating un-calibrated values of $\alpha_{ij}$ in the loss, we propose removing the $K$-nearest neighbours from the set of negative samples based on $\alpha_{ij}$. To this end, for each postive sample $\mathbf{q}_i$, we sort $\alpha_{ij}$, $\forall j \neq i$ and compute $\alpha_{iK}$ that contains the $K$-nearest neighbours. Here, $C_{ij}$ is an indicator of whether $\mathbf{f}_j$ is semantically similar to $\mathbf{f}_i$.

$$C_{ij} = \begin{cases} 1.0, & \text{if } \alpha_{ij} < \alpha_{iK} \\ 0.0, & \text{otherwise} \end{cases} \tag{5}$$

The loss can then be formulated as:

$$\mathcal{L}_{knn}\left(\mathbf{Q},\mathbf{K}\right) = \\ -\frac{1}{M}\sum_{i=0}^{M}\log\left[\frac{e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}{\sum_{j\neq i} C_{ij} \;.\; e^{(\langle \mathbf{q}_i,\mathbf{k}_j\rangle/\tau)} + e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}\right]. \tag{6}$$

Using $\mathcal{L}_{knn}$, a fixed number of negative samples, excluding the $K$-nearest neighbors are used as the negative set. We show in Table 6 that excluding the $K$-nearest neighbors results in better linear separability compared to directly incorporating $\alpha_{ij}$ in the contrastive loss.

### 3.3. Class-Agnostic Balanced Contrastive Loss

Autonomous driving datasets are highly imbalanced (see Figure 2), for instance, in nuScenes [6] only $0.05\%$ of the superpixels belong to 'motorcycle' and 'bicycle' categories, while $45\%$ of the superpixels belong to 'road' and 'vegetation' categories. For indoor 3D point cloud datasets, the problem is less severe, where the rarest category 'sink' in ScanNet V2 [10] consists of $2.75\%$ of the points [15]. Since the category of a sample is unknown during pretraining, PPKT [17] and SLidR [21] assume a fixed weight of $\frac{1}{M}$ on the loss from each sample within a batch. Since the training signal is dominated by gradients from samples of over-represented categories, the learned representations for under-represented categories might not be optimal.

To address this issue, we reason that superpixel-to-superpixel similarity can also be used as a proxy for class imbalance. For example, for an over-represented anchor $\mathbf{q}_i$ in a batch, its associated superpixel feature $\mathbf{f}_i$ will have high $\alpha_{ij}$ with a large number of negative samples, while an under-represented anchor, will have low $\alpha_{ij}$ with most negative samples. To balance the training, first, we compute votes for each anchor based on similarity $v_i = \sum_{j=1}^{M}\alpha_{ij}$. Then, a min-max normalization is applied $v_i = \frac{v_i - v_{min}}{v_{max}}$ to suppress noise. Finally, for each anchor $\mathbf{q}_i$, we assign a weight $w_i$ inversely proportional to $v_i$. The semantically-tolerant loss with class-agnostic balancing can be formulated as:

$$\mathcal{L}_{ST}\left(\mathbf{Q},\mathbf{K}\right) = \\ -\sum_{i=0}^{M}\frac{w_i}{w}\log\left[\frac{e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}{\sum_{j\neq i} C_{ij} \;.\; e^{(\langle \mathbf{q}_i,\mathbf{k}_j\rangle/\tau)} + e^{(\langle \mathbf{q}_i,\mathbf{k}_i\rangle/\tau)}}\right]. \tag{7}$$

Where $w_i = 1.0 - v_i$ and $w = \sum_{i=0}^{M} w_i$.

## 4. Experiments

### 4.1. Pre-training

**Backbones**. To represent the input point cloud, we transform the 3D points from Cartesian coordinates $(x, y, z)$ to cylindrical coordinates $(\rho, \phi, z)$. The input is voxelized using 3D cylindrical partitioning [30], where the voxel size is $(\delta\rho = 10cm, \delta\phi = 1^\circ, \delta z = 10cm)$. For the 3D backbone, similar to SLidR [21], we use the Minkowski U-Net with $3 \times 3 \times 3$ kernels for all sparse convolutional layers. For the 2D backbone, we use the ResNet-50 [12] architecture and initialize the model parameters using a multitude of 2D self-supervised pretrained models including MoCoV2 [10], SwAV [7], DINO [8], OBoW [11] and DenseCL [25]. For all experiments except 2D SSL frameworks, the 2D backbone for PPKT [17], SLidR [21] and ST-SLidR is initialized using MoCoV2 [10].

**Dataset**. For pre-training, we use the nuScenes [6] dataset, which contains 700 training scenes. Following SLidR [21], we further split the 700 scenes into 600 for pre-training and 100 scenes for selecting the optimal hyper-parameters. During pretraining, only keyframes from the 600 scenes are used to train both SLidR and ST-SLidR.

**Training and Data Augmentations**. For all experiments, we pre-train the point cloud encoder, $f_{\theta_P}$, the superpoint embedding layer, $h_{\omega_P}$ and the superpixel embedding layer, $h_{\omega_I}$, for 50 epochs on 2 A100 GPUs with a batch size of 16. Similar to SLidR [21], we use an SGD optimizer with a momentum of 0.9, an initial learning rate of 0.5 and a cosine annealing learning rate scheduler. Finally, for regularization, we use a weight decay of 0.0001 and dampening of 0.1. For data augmentation, we apply linear transformations to the point cloud including random flipping in the $x$ and $y$-axis, and rotating around $z$-axis. In addition, similar to DepthContrast [29], we randomly select a cube and drop all points within the cube. For images, we apply random crop-resize and horizontal flip.

### 4.2. Evaluation

To assess the quality of the pre-trained 3D representations, a point-wise linear classifier is added to the output of $f_{\theta_P}$. Two protocols are used to evaluate the performance on

| 3D<br>Initialization | Reference | nuScenes<br>Lin. Prob<br>100% | <br>Finetune<br>1% | KITTI<br>Finetune<br>1% | Waymo<br>Finetune<br>1% |
|---|---|---|---|---|---|
| Random | N/A | 8.10 | 30.30 | 39.50 | 39.41 |
| PointContrast [27] | ECCV 2020 | 21.90 | 32.50 | 41.10 | - |
| DepthContrast [29] | ICCV 2021 | 22.10 | 31.70 | 41.50 | - |
| PPKT [17] | arVix 2021 | 35.90 | 37.52 | 44.00 | - |
| SLidR [21] | CVPR 2022 | 38.40 | 38.83 | 43.96 | 47.12 |
| ST-SLidR | - | **40.48** | **40.75** | **44.72** | **47.93** |
| *Improvement* | | *+2.08* | *+1.92* | *+0.76* | *+0.81* |

Table 1. Semantic segmentation results on nuScenes [6], SemanticKITTI [5] and Waymo [22] validation sets using 3D self-supervised methods. On nuScenes, we evaluate linear probing using 100% of the annotated training set and for finetuning, we evaluate using 1% of the data. In addition, we evaluate out-of-distribution performance on SemanticKITTI and Waymo datasets using 1% of the training set. All models are pretrained using the nuScenes dataset.

a semantic segmentation task, linear probing [2], and finetuning. For linear probing, the parameters of $f_{\theta_P}$ are frozen and only the classifier head is trained on 100% of the training data from the nuScenes dataset. Since gradients are not propagating back to $f_{\theta_P}$, the linear probe protocol directly evaluates the quality of the pre-trained representations. We evaluate the performance of the linear probing protocol on the nuScenes validation dataset.

For fine-tuning, the representations are evaluated under a low sample count, where the model is finetuned end-to-end using only 1% of the annotated training data. The finetuning protocol allows us to study the utility of the pre-trained representations under a limited annotation budget. We study the finetuning performance on the nuScenes validation dataset and also evaluate the utility of the learned pre-trained nuScenes representations during finetuning on out-of-distribution data from the SemanticKITTI [5] and Waymo [22] datasets. The number of classes for nuScenes, SemanticKITTI and Waymo datasets are 16, 19 and 22 respectively. The results are reported on the official validation sets of all datasets. For training, we use a linear combination of the Lovasz and cross-entropy loss, and the same training hyperparameters as SLidR.

## 4.3. Results

**Comparison with Baselines**. In Table 1, we present the performance of Random initialization, PointContrast [27], and DepthContrast [29] reported in [21]. For PPKT [17], SLidR [21] and ST-SLidR, we run 3 pre-training experiments using SLidR's code base and report the mean performance for each setting. We observe that models initialized using weights from 3D SSL frameworks provide significant improvements over random initialization. In addition, in an outdoor setting where the density of the point cloud falls off rapidly at mid-to-long range, 3D SSL methods using superpixel-driven loss like SLidR [21] leads to improved performance compared to point-level losses like PointContrast [27] and PPKT [17] or scene-level losses like DepthContrast [29]. Pre-training using our proposed semantically-tolerant and balanced loss ST-SLidR provides a significant mIoU improvement of +2.08% for linear probing and +1.92% for few-shot fine-tuning tasks over state-of-the-art SLidR on nuScenes datasets. In addition, compared to SLidR, ST-SLidR also achieves better generalization in out-of-distribution few-shot semantic segmentation on SemanticKITTI and Waymo datasets. We also show that our proposed loss can improve the quality of 3D representations of pixel-to-point contrastive losses such as PPKT [17] (see Table 7 in Appendix)

| 3D<br>Initialization | 2D<br>Initialization | nuScenes<br>Lin. Prob<br>100% | <br>Finetune<br>1% | KITTI<br>Finetune<br>1% |
|---|---|---|---|---|
| SLidR<br>ST-SLidR | MoCoV2 [10] | 38.40<br>**40.48** | 38.83<br>**40.75** | 43.96<br>**44.72** |
| *Improvement* | | *+2.08* | *+1.92* | *+0.76* |
| SLidR<br>ST-SLidR | SwAV [7] | 39.11<br>**40.49** | 38.81<br>**40.86** | 44.15<br>**44.98** |
| *Improvement* | | *+1.38* | *+2.05* | *+0.83* |
| SLidR<br>ST-SLidR | DINO [8] | 37.76<br>**40.31** | 38.57<br>**40.52** | 43.94<br>**44.24** |
| *Improvement* | | *+2.55* | *+1.95* | *+0.30* |
| SLidR<br>ST-SLidR | OBoW [11] | 36.56<br>**39.81** | 38.55<br>**40.08** | 43.84<br>**44.58** |
| *Improvement* | | *+3.25* | *+1.53* | *+0.74* |
| SLidR<br>ST-SLidR | DenseCL [25] | 34.90<br>**36.93** | 37.50<br>**39.21** | **43.44**<br>43.34 |
| *Improvement* | | *+2.03* | *+1.71* | *-0.10* |

Table 2. The performance of vanilla SLidR compared to our proposed loss (ST-SlidR) when using different pretrained 2D self-supervised models during training.

**2D SSL Frameworks**. In Table 2, we present results for experiments using weights pre-trained with different 2D SSL frameworks. We observe that ST-SLidR provides significant improvements over SLidR of at least 1.38% using linear probing and 1.5% for fine-tuning on nuscenes dataset across all 2D pretrained models. This shows the robustness of ST-SLidR to selection of the 2D SSL pretrained model, indicating real benefit to feature transfer from 2D to 3D point encoders.

**Annotation Efficiency**. In Table 3 we present results on the utility of the pre-trained representations as a function of the percentage of nuScenes training set. Here, we use the same training parameters selected by SLidR [21] including learning rate and number of training epochs to evaluate the se-

| 3D Initialization | 1% | 5% | 10% | 25% | 100% |
|---|---|---|---|---|---|
| Random | 28.64 | 47.84 | 56.15 | 65.48 | 74.66 |
| SLidR | 38.83 | 52.49 | 59.84 | 66.91 | 74.79 |
| ST-SLidR | **40.75** | **54.69** | **60.75** | **67.70** | **75.14** |
| *Improvement* | *+1.92* | *+2.20* | *+0.91* | *+0.79* | *+0.35* |

Table 3. Finetune results on nuScenes as a function of the percentage of annotated data. Improvements are shown with respect to SLidR [21]

| Method | Lin. Prob 100% | | Finetune 1% | |
|---|---|---|---|---|
| | min | maj | min | maj |
| SLidR | 27.46 | **62.47** | 22.96 | 73.75 |
| ST-SLidR | **30.64** | 62.15 | **25.56** | **74.15** |
| *Improvement* | *+3.18* | *-0.32* | *+2.61* | *+0.40* |

Table 4. We report mIOU for minority (min) and majority (maj) classes of nuScenes dataset. We group classes based on whether their superpixels occupy more than **5%** of the superpixels in nuScenes training set.

| Sematic Tolerant Loss | Class Balanced Loss | Lin. Prob 100% | Finetune 1% |
|---|---|---|---|
| ✗ | ✗ | 37.87 | 38.96 |
| ✗ | ✓ | 38.33 | 39.73 |
| ✓ | ✗ | 40.04 | 40.19 |
| ✓ | ✓ | **40.48** | **40.75** |

Table 5. Contribution of semantic awareness and class agnostic balancing on ST-SLidR.

| Loss | Lin. Prob 100% MoCoV2 | Lin. Prob 100% SwAV |
|---|---|---|
| $\alpha_{min} = 0.0$ | 37.99 | 36.21 |
| $\alpha_{min} = 0.2$ | 38.64 | 36.86 |
| $\alpha_{min} = 0.5$ | 39.48 | 40.03 |
| $\alpha_{min} = 0.8$ | 38.23 | 39.15 |
| $K = 1\%$ | 40.04 | 40.42 |
| $K = 5\%$ | **40.38** | **40.84** |
| $K = 10\%$ | 40.35 | 40.05 |

Table 6. Comparison between similarity-aware $\mathcal{L}_{\alpha}$ versus nearest-neighbour-aware $\mathcal{L}_{knn}$ loss. Here, we report mIOU on the validation set of nuScenes [6] set.

mantic segmentation fine-tuning performance of SLidR and ST-SLidR as a function of the number of labelled scenes. We observe that ST-SLidR outperforms SLidR by +1.92%, +2.20%, +0.91%, +0.79% and +0.35% when fine-tuning on 1%, 5%, 10%, 25% and 100% of the dataset, respectively.

**Class Imbalance**. We conduct an experiment to study which semantic classes gain the most from the semantically tolerant contrastive loss. We compute the percentage of superpixels for each semantic class in the nuScenes pre-training set. Then, we create two sets of classes. The minority set contains all classes with fewer than 5% of the superpixels in the pre-training set. The remaining classes are added to the majority set. Out of the 16 classes in the nuScenes dataset, 11 classes are categorized as minority classes. In Table 4, the mean IoU for minority and majority sets for linear probing and fine-tuning on nuScenes validation set is presented. Compared to SLidR, ST-SLidR learns representations that significantly improve segmentation performance by +3.18% for linear probing and +2.61% for fine-tuning on the 11 minority classes. Interestingly, the significant improvement on minority classes comes with almost no degradation on majority classes.

### 4.4. Ablations

**Contribution of Loss Components**. We conduct ablation studies to validate the contribution of the two components of ST-SLidR. Here, semantic tolerant loss denotes $\mathcal{L}_{knn}$ with K set to 1% of the mini-batch. Table 5 shows that (1) Both semantic tolerant and class balancing can improve the quality of the learned representation on their own, (2) Semantic tolerant loss significantly improves the linear separability of the 3D representations as fewer false negative samples contribute to the loss, (3) Using both components achieves the best performance on linear probing and few-shot semantic segmentation.

**Similarity versus Nearest-Neighbour-aware Loss**. We conduct an ablation study to show the quality of the 3D representations for similarity-aware loss $\mathcal{L}_{\alpha}$ and nearest neighbour-aware loss $\mathcal{L}_{knn}$. For pre-training experiments using $\mathcal{L}_{\alpha}$, we vary the minimum similarity threshold $\alpha_{min}$ and for $\mathcal{L}_{knn}$, we vary percentage of top-$K$ nearest neighbours to be excluded from the set of negative samples. Table 6 shows pre-training with $\mathcal{L}_{knn}$ results in 3D representations that are much more linearly separable than $\mathcal{L}_{\alpha}$.

## 5. Conclusion

We present a novel 2D-to-3D representation learning framework for autonomous driving datasets that reduces the contribution of false negative samples by explicitly considering the similarity of self-supervised image features. In addition we propose balancing the pretraining between over and under-represented samples by using aggregate sample-to-samples similarity as a proxy for class imbalance. Our proposed contributions are shown to additively improve common 2D-to-3D representation learning methods in all evaluation settings on 3D semantic segmentation, especially for under-represented classes.

## References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE TPAMI*, 34(11):2274–2282, 2012.

[2] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

[3] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *NeurIPS*, 32, 2019.

[4] Adrien Bardes, Jean Ponce, and Yann Lecun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR*, 2022.

[5] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *ICCV*, 2019.

[6] 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 *CVPR*, pages 11621–11631, 2020.

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 33:9912–9924, 2020.

[8] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pages 9650–9660, 2021.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607, 2020.

[10] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[11] Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Perez. Obow: Online bag-of-visual-words generation for self-supervised learning. In *CVPR*, pages 6830–6840, 2021.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[13] Olivier J Hénaff, Skanda Koppula, Jean-Baptiste Alayrac, Aaron Van den Oord, Oriol Vinyals, and João Carreira. Efficient visual pretraining with contrastive detection. In *ICCV*, pages 10086–10096, 2021.

[14] Jordan S. K. Hu, Tianshu Kuai, and Steven L. Waslander. Point density-aware voxels for lidar 3d object detection. In *CVPR*, pages 8469–8478, June 2022.

[15] Li Jiang, Shaoshuai Shi, Zhuotao Tian, Xin Lai, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Guided point contrastive learning for semi-supervised point cloud semantic segmentation. In *ICCV*, pages 6423–6432, 2021.

[16] Lanxiao Li and Michael Heizmann. A closer look at invariances in self-supervised pre-training for 3d vision. *ECCV*, 2022.

[17] Yueh-Cheng Liu, Yu-Kai Huang, Hung-Yueh Chiang, Hung-Ting Su, Zhe-Yu Liu, Chin-Tang Chen, Ching-Yu Tseng, and Winston H Hsu. Learning from 2d: Contrastive pixel-to-point knowledge transfer for 3d pretraining. *arXiv preprint arXiv:2104.04687*, 2021.

[18] Anas Mahmoud, Jordan S. K. Hu, and Steven L. Waslander. Dense voxel fusion for 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 663–672, January 2023.

[19] Anas Mahmoud and Steven L. Waslander. Sequential fusion via bounding box and motion pointpainting for 3d objection detection. In *CRV*, 2021.

[20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

[21] Corentin Sautier, Gilles Puy, Spyros Gidaris, Alexandre Boulch, Andrei Bursuc, and Renaud Marlet. Image-to-lidar self-supervised distillation for autonomous driving data. In *CVPR*, pages 9891–9901, 2022.

[22] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, pages 2446–2454, 2020.

[23] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[24] Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *CVPR*, pages 2495–2504, 2021.

[25] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *CVPR*, pages 3024–3033, 2021.

[26] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018.

[27] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *ECCV*, pages 574–591. Springer, 2020.

[28] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, pages 12310–12320. PMLR, 2021.

[29] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. Self-supervised pretraining of 3d features on any point-cloud. In *ICCV*, pages 10252–10263, October 2021.

[30] 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 *CVPR*, pages 9939–9948, 2021.