From One to the Power of Many: Invariance to Multi-LiDAR Perception from Single-Sensor Datasets

Marc Uecker[1], **J. Marius Zöllner**[1,2]

[1]FZI Research Center for Computer Science
[2]Karlsruhe Institute of Technology
uecker@fzi.de

Abstract

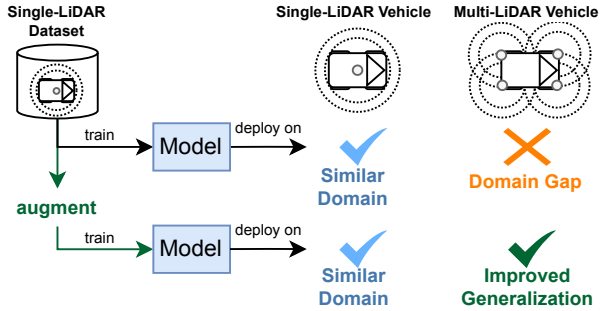
Recently, LiDAR segmentation methods for autonomous vehicles, powered by deep neural networks have experienced steep growth in performance on classic benchmarks, such as nuScenes and SemanticKITTI. However, there are still large gaps in performance when deploying models trained on such single-sensor setups to modern vehicles with multiple high-resolution LiDAR sensors. In this work, we introduce a new metric for feature-level invariance which can serve as a proxy to measure cross-domain generalization without requiring labeled data. Additionally, we propose two application-specific data augmentations, which facilitate better transfer to multi-sensor LiDAR setups, when trained on single-sensor datasets. We provide experimental evidence on both simulated and real data, that our proposed augmentations improve invariance across LiDAR setups, leading to improved generalization.

1 Introduction

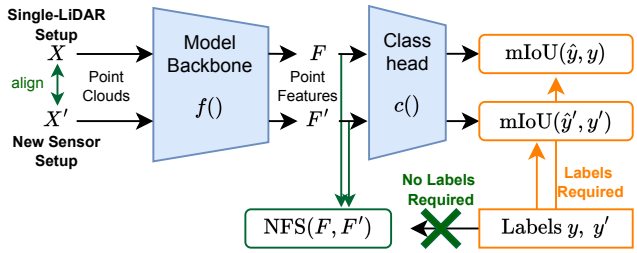
LiDAR sensors are an essential component of autonomous vehicle technology, commonly used for 3D environment perception and localization. Many modern research prototypes (Karle et al. 2023; Heinrich et al. 2024) and automated vehicles deployed in real traffic (Ayala and Mohd 2021) include multiple LiDAR sensors as part of their sensor suite. However, datasets commonly used for training 3D semantic segmentation models typically only provide annotated data from a single LiDAR sensor (Behley et al. 2019; Caesar et al. 2020). Models trained on these datasets usually do not work well out-of-the-box on the fused point clouds of multiple LiDAR sensors, as illustrated in figure 1(a) (top).

Since annotating 3D point clouds for re-training is prohibitively expensive, the main goal of this work is to enable zero-shot generalization from single-sensor datasets to multi-sensor setups in modern vehicles. We aim to achieve this by introducing two augmentations for improving the invariance of segmentation models to changes in sensor setup (see figure 1(a), bottom). Even with robust models, there remains a need to evaluate their generalization capabilities without access to labels. To facilitate this, we introduce a new metric called *Normalized Feature Similarity* (NFS), which serves as a proxy for generalization performance, and can be applied on real-world data without requiring labels.

Workshop on Machine Learning for Autonomous Driving at AAAI 2025 (ML4AD@AAAI 2025), Philadelphia, USA. ml4ad.github.io



(a) We introduce two new augmentations for single-sensor datasets which improve zero-shot generalization to multi-sensor vehicles.



(b) We establish *Normalized Feature Similarity* (NFS) as a proxy metric for out-of-domain generalization performance (e.g. mIoU score), since it can be applied on un-labeled data.

Figure 1: We tackle the problem of zero-shot generalization to unseen data from modern multi-LiDAR vehicles, for which labels are not available.

We demonstrate on simulated data that our NFS metric correlates well with out-of-domain segmentation performance.

Key Contributions

- We improve the zero-shot generalization of LiDAR segmentation models to multi-sensor vehicles by introducing two new data augmentations which can be applied on single-sensor datasets.
- We propose a new *Normalized Feature Similarity* (NFS) metric to quantify feature-level invariance without requiring labels, and show that NFS empirically correlates well with out-of-domain segmentation performance.
- Using this NFS metric, we show that our augmentations increase invariance to sensor setup changes on labeled simulated data as well as real-world data without labels.

2 Related Work

In this section, we give a brief overview of relevant works related to invariance and augmentations for 3D point clouds.

Invariance in Point Cloud Semantic Segmentation Semantic segmentation of LiDAR point clouds is a task that inherently includes many symmetries. As a set of 3D points, LiDAR point clouds do not necessarily have a preferred order, and re-ordering the points does not change the class of the individual points. Therefore, permutation invariance (i.e. the features of points being unaffected by their order) becomes a desirable property in models for LiDAR semantic segmentation (Kimura et al. 2024), and is a common feature of state-of-the-art architectures (Zaheer et al. 2017; Zhao et al. 2021). Another common target for invariances is the group of 3D translations and rotations, commonly named SE(3). Many architectures for 3D semantic segmentation feature at least partial invariance to either translations or rotations in special frames of reference (Qi et al. 2017; Liu et al. 2019; Zhu et al. 2021; Uecker et al. 2022). However, architectures with fully built-in SE(3) invariance typically incur high computation time and memory costs (Chen et al. 2021), which limits their adoption in real-time applications like autonomous driving. Fang et al. (2024) show that common LiDAR object detection models are not very robust to changes in the LiDAR sensor's scan pattern, but there is not yet any method against this.

Augmentations and Invariances Empirically, the use of data augmentation during training can induce approximate invariances, i.e. cases where for a class of data transformations t, a model f might gain properties such that $f(t(x)) \approx f(x)$. Multiple recent works explore these notions of approximate invariance of neural networks in further detail (Bouchacourt, Ibrahim, and Morcos 2021; Kvinge et al. 2022; Botev, Bauer, and De 2022). In this work, we lean on this understanding of invariance as a metric, rather than a baked-in property of architectures such as group-equivariant (Cohen and Welling 2016) or steerable convolutions (Weiler, Hamprecht, and Storath 2018). This view of invariance permits us to use an existing state-of-the-art architecture, and quantitatively measure changes in invariance behavior.

Augmentations for 3D Point Clouds State-of-the-Art models for LiDAR segmentation are often trained using numerous data augmentations, such as mirroring, and translation and rotation at both object and scene level (Li et al. 2020). We use these as our baseline set of augmentations. Replacing or mixing objects and scenes from a dataset is also a common strategy (Xiao et al. 2022).

3 Feature Invariance as a Label-free Proxy for Out-of-Domain Generalization

While performance metrics such as mIoU can directly measure whether a trained model generalizes to a different sensor setup, they require expensive annotated data (with matching class definitions) for each application domain. In order to avoid this need for annotated data, we introduce a metric called *Normalized Feature Similarity* (NFS) in this section. Our NFS metric quantifies how invariant a trained model is to changes in sensor setups. We design NFS as a proxy for classical segmentation metrics, for situations where labeled data is not available. We empirically verify that our NFS metric correlates well with out-of-domain mIoU scores in section 3.2 using simulated data.

3.1 Normalized Feature Similarity

To quantify the invariance of a trained model, we use a feature similarity method by Kvinge et al. (2022) to compare feature vectors of points across different sensor setups, as illustrated in figure 1(b). Given two aligned point clouds $X, X' \in \mathbb{R}^{N\times 3}$, where $X = \{\mathbf{x}_i \in \mathbb{R}^3 \mid i \in 1..N\}$, along with point-wise features $F, F' \in \mathbb{R}^{N\times d}$, where $F = f(X) = \{\mathbf{f}_i \in \mathbb{R}^d \mid i \in 1..N\}$ and $F' = f(X')$, computed by the same feature extractor backbone function $f : \mathbb{R}^{N\times 3} \to \mathbb{R}^{N\times d}$, we can compute the cosine similarity between the features of each point:

$$\mathrm{sim}(\mathbf{f}_i, \mathbf{f'}_i) = \frac{\langle \mathbf{f}_i, \mathbf{f'}_i \rangle}{||\mathbf{f}_i||_2 \, ||\mathbf{f'}_i||_2} \tag{1}$$

where $\langle \cdot, \cdot \rangle$ is the scalar product, as proposed by Kvinge et al. for image features. However, the features extracted by neural networks can be arbitrarily shifted and scaled through different weights and biases. A large bias value in the previous layer will lead to high similarities regardless of content. We therefore propose to normalize the features before computing cosine similarity values. We compute feature-wise mean $\boldsymbol{\mu}_F \in \mathbb{R}^d$ and standard deviation $\boldsymbol{\sigma}_F \in \mathbb{R}^d$ across the features F to normalize the distribution of each feature and remove model-to-model variations in feature scales and shifts. Therefore, our proposed *Normalized Feature Similarity* (NFS) is computed as:

$$\mathrm{NFS}(F, F') = \frac{1}{N} \sum_{i=1}^{N} \mathrm{sim}\Big(\frac{\mathbf{f}_i - \boldsymbol{\mu}_F}{\boldsymbol{\sigma}_F}, \frac{\mathbf{f'}_i - \boldsymbol{\mu}_F}{\boldsymbol{\sigma}_F} \Big) \tag{2}$$

Since we want to quantify the invariance of a single trained model f, we only compare feature vectors by the same model with identical weights. Therefore, more sophisticated methods to account for different model weights, e.g by matching channel permutations between models (Li et al. 2015), are not required in this work.

Evaluation across Sensor Setups In order to compare features across different LiDAR sensor setups, our NFS metric requires *aligned* pairs of point-wise feature vectors ($\mathbf{f}_i$ and $\mathbf{f'}_i$ in eq. (2)). To facilitate this, we match each LiDAR point $\mathbf{x}'_i$ (and its features) from a new setup to the closest point $\mathbf{x}_i$ with the same timestamp from the single-LiDAR setup using a nearest-neighbor search across X, with a search radius of 1 meter. Points $\mathbf{x}'_i$ with no corresponding neighbors in X within this radius (usually due to limited FOV overlap between setups, shown as blank spaces in figure 6(b) and 6(c) second from left) are ignored for our evaluation.

3.2 Results on Simulated Data

In order to verify that our NFS metric is meaningful with regards to predicting out-of-domain generalization performance, we train and evaluate various models on simulated LiDAR data.

We use a sparse point-voxel CNN (Tang et al. 2020) architecture for all of our experiments, since it permits using multi-sensor point clouds, which would cause issues in a range-view representation (Triess et al. 2020).

Simulations Inspired by Fang et al. (2024), we generate simulated data in CARLA (Dosovitskiy et al.) for 19-class semantic segmentation, using different sensor setups in the same environment to isolate and investigate the effects of sensor setup changes to model performance. First, we simulate an in-domain setup similar to the SemanticKITTI dataset for training, with a single 64-channel 360° LiDAR placed centrally on the vehicle roof. Then we simulate various out-of-domain sensor setups for evaluation. Our simulations capture a standardized scenario spanning 5.56 hours of simulated time (10,000 sensor time steps at 0.5 Hz) of driving through procedurally generated traffic, which is re-simulated with various different sensor setups. The training set consists of the last 6,000 time steps. A test set consisting of the first 2,000 steps is held for evaluations.

Evaluation We train a variety of models with different augmentations on the single-sensor in-domain setup. We then evaluate the NFS and mIoU scores on the test set of many sensor setup variations, including: 1. The sensor's horizontal FOV (60°-360° in 60° increments). 2. The number of sensors (up to four sensors mounted on the corners of the roof), each with 64 channels. 3. The number of sensors n (up to four), each with $64/n$ channels for a constant number of points per point cloud. 4. The sensor's vertical resolution (# of channels in the single-LiDAR setup).

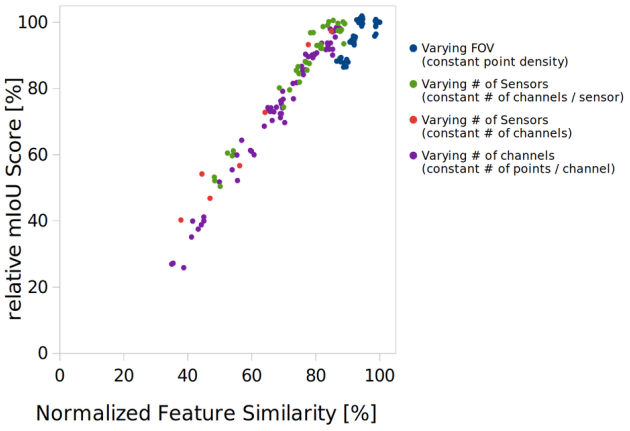
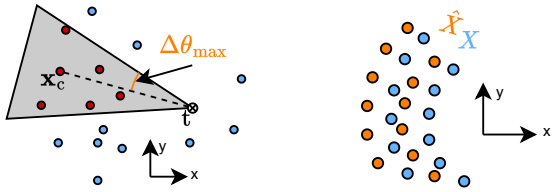


Figure 2: Correlation between our Normalized Feature Similarity metric and out-of-distribution relative mIoU Score (rmIoU, as a percentage of in-domain test set mIoU) for a variety of models and sensor setups.

Figure 2 shows a scatter plot of relative mIoU score (rmIoU) over our Normalized Feature Similarity across all listed sensor setup variations, each evaluated with multiple models trained with different augmentation configurations. A linear regression yields $\mathrm{rmIoU}[\%] = 1.04 * \mathrm{NFS}[\%] + 1.63[\%]$ with $R^2 = 0.916$, suggesting that in the absence of labels, our NFS metric can be a very good proxy for out-of-domain generalization.

4 Augmentations for Invariance to Sensor Setup Changes

In order to improve the zero-shot out-of-domain generalization of our semantic segmentation models, we propose two augmentations specifically designed to mimic the effects of sensor setup changes.



(a) Frustum Drop: Points within a randomly sampled view frustum (gray) are dropped, in order to imitate occlusions and limited sensor field-of-view.

(b) Mis-Calibration Augmentation: The input point cloud X is duplicated, shifted, and slightly rotated to imitate overlapping scans from multiple sensors.

Figure 3: The two augmentations proposed in this work

4.1 Frustum Drop Augmentation

In order to make our models more robust to occlusions and changes in field of view, we design an augmentation, which drops points from a randomly selected view frustum from the point cloud. This is the inverse opresentf *frustum culling*, an operation commonly used in 3D game engines, but can also be thought of as a variation of CutOut augmentation (DeVries 2017) on a spherical projection of the point cloud onto a randomly chosen origin point. Figure 3(a) illustrates this augmentation.

Given a point cloud $X \in \mathbb{R}^{N\times 3} = \{\mathbf{x}_i \in \mathbb{R}^3 \mid i \in 1..N\}$, our augmentation is performed as follows: We first sample a random origin coordinate $\mathbf{t} \in \mathbb{R}^3$, $\mathbf{t} \sim [-r, r]^3$ as the origin of the frustum from a uniform distribution, where r determines the size of a cubic region being sampled from. We use a value of 3 meters for r. Then, we randomly choose one of the points $\mathbf{x}_c$ from the point cloud X as the center of the omitted frustum. We then compute azimuth angles θ_i and corresponding elevation angles ψ_i for each point $\mathbf{x}_i$ relative to the frustum origin $\mathbf{t}$, using the following equations:

$$\theta_i = \mathrm{atan2}\left(\frac{\hat{y}}{\hat{x}}\right), \psi_i = \mathrm{atan2}\left(\frac{\hat{z}}{\sqrt{\hat{x}^2 + \hat{y}^2}}\right), \begin{pmatrix}\hat{x}\\ \hat{y}\\ \hat{z}\end{pmatrix} = (\mathbf{x}_i - \mathbf{t})$$

Using our sampled point $\mathbf{x}_c$ as the center of the frustum, we compute the angles $\Delta\theta_i = \arccos(\cos(\theta_i - \theta_c))$ and $\Delta\psi_i = \arccos(\cos(\psi_i - \psi_c))$ of each point $\mathbf{x}_i$ relative to $\mathbf{x}_c$. The combination of $\arccos$ and $\cos$ normalizes the angle difference to a positive value in the range of $[0, 180°]$. Finally, we drop all points $\mathbf{x}_i$ whose relative angles $\Delta\theta_i$ and $\Delta\psi_i$ are both within a range $[0, \Delta\theta_{\max}]$ and $[0, \Delta\psi_{\max}]$ respectively, where we sample $\Delta\theta_{\max}$ and $\Delta\psi_{\max}$ uniformly from the range $[2.5°, 90°]$. A new combination of parameters ($\mathbf{t}$, j, $\Delta\theta_{\max}$, and $\Delta\psi_{\max}$) is sampled randomly for each new point cloud. In effect, this augmentation removes at least one point, and up to half the field of view from the point cloud in a pyramid cone shape.

Augmentation	Test mIoU ↑ (in-domain)	Zero-shot mIoU ↑ (out-of-domain, 64 channels) 1 Sensor	2 Sensors	3 Sensors	4 Sensors	Zero-shot mIoU ↑ (Varying # of Channels) 16	32	48	64	96	128	192	256
Base	76.1	67.8	56.6	45.4	38.4	19.7	46.6	65.2	76.1	68.6	57.4	45.6	39.7
Base + FD(p=0.5)	75.8	**75.9**	60.3	46.4	40.3	20.6	48.8	65.7	75.8	68.5	56.2	46.3	42.0
Base + FD(p=1)	75.8	75.5	60.7	45.8	39.5	20.4	45.5	64.6	75.8	67.9	56.2	45.4	39.2
Base + MC(p=0.125, s=0.05)	**77.6**	72.5	71.5	66.4	63.6	**31.0**	54.1	69.9	**77.6**	**74.1**	69.3	59.6	55.2
Base + MC(p=0.25, s=0.05)	75.3	73.6	70.3	66.2	64.3	**31.0**	54.9	70.6	75.3	73.7	68.0	59.6	55.3
Base + MC(p=0.5, s=0.05)	76.3	74.4	70.9	67.2	66.0	30.5	55.7	**70.9**	76.3	**74.1**	69.2	62.4	**58.5**
Base + MC(p=0.5, s=1.0)	74.9	72.9	75.0	73.9	**72.5**	26.3	55.5	68.7	74.9	73.7	**70.2**	61.1	52.7
Base + FD(p=0.5) + MC(p=0.5, s=1.0)	74.9	74.5	**75.3**	**74.1**	**72.5**	29.1	**55.6**	68.7	74.9	73.5	70.1	**63.0**	54.2

Table 1: Generalization performance (zero-shot mIoU Score) of different augmentation configurations on the unseen test split of the simulated out-of-domain *Corner* sensor setups shown in Figure 6(a) (1-4 sensors) and the different sensor resolutions from 16 to 256 channels. Each row shows a single model, trained on the training split of the single-LiDAR sensor setup.

4.2 Mis-Calibration Augmentation

When combining point clouds from multiple sensors, their individual field-of-views often overlap significantly, leading to higher point density, and overlapping scan line artifacts which don't appear in single-sensor point clouds (compare illustrations figure 1(a) top center vs top right). Since our aim is to train on existing single-sensor datasets, these effects are not represented in our training data. We propose a *Mis-Calibration Augmentation* to artificially introduce these effects in single-sensor training data. By duplicating the entire point cloud, we can create a point cloud with twice the local point density everywhere.

By applying a slight random rotation and translation to one of the copies, we can create additional effects resembling a slightly mis-calibrated pair of sensors with the same scan pattern. We visualize this augmentation in Figure 3(b). More formally, we compute the copied point cloud $\hat{X}$ as:

$$\hat{X} = (X \cdot R^\top) + \mathbf{t}^\top \qquad R = R_z(\alpha_z)R_y(\alpha_y)R_x(\alpha_x)$$

$$\mathbf{t} \sim [-s_{xy}, s_{xy}]^2 \times [-s_z, s_z] \quad \alpha_x, \alpha_y, \alpha_z \sim [-\alpha_{\max}, \alpha_{\max}]$$

where $R_z(\alpha)$ denotes a rotation around the z-axis with angle α. Our augmentation typically generates new points which are slightly offset from the original surface originally sampled by the LiDAR sensor. Therefore, we attempt to limit the offset caused by our augmentation to values that we assume to be similar to sensor noise. Therefore, we set s_{xy}, s_z and $\alpha_{\max}$ to small values of 0.05 m and 0.05° respectively. We also experiment with a higher value of 1 m for s_{xy}, and find that in-domain performance is barely impacted by this change. Since this augmentation causes an increase in the overall point density seen during batch training, we only apply it with a probability of up to 50% during training, as we reason that higher probabilities may cause our model to lose performance on the original in-domain setup.

4.3 Results on Simulated Data

Experiment details Using the same simulations and training procedure as in section 3.2, we evaluate both mIoU score and NFS metric for our newly introduced augmentations. Hereby, we train models with a strong baseline set of SE(3) augmentations, as well as adding one or both of our proposed augmentations on the simulated single-LiDAR in-domain setup. We then evaluate the zero-shot out-of-domain mIoU score, as well as NFS relative to the single-LiDAR in-domain setup for each of these models on the test set of multiple new sensor setups.

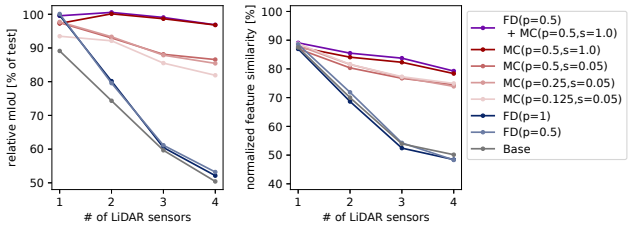


Figure 4: Comparing relative mIoU score (left) and NFS (right) of our proposed augmentations on the test partition of the simulated out-of-domain *Corner* sensor setups shown in Figure 6.

Evaluation across Single-Sensor Setups Table 1 (left) shows the mIoU scores of these trained models when applied to various out-of-domain configurations, where each sensor has 64 channels, as in the in-domain setup. Without our proposed augmentations, the *Base* model achieves a good in-domain performance, but loses performance when deployed on a different single-sensor setup (corner instead of center of roof). As seen in table 1 (second column), the different placement and horizontal FOV already reduces the mIoU score by 8 points. Our Frustum Drop augmentation (table 1, rows 2 and 3) alleviates this issue, and achieves an almost identical mIoU score on both single-sensor setups (first 2 columns). Our Mis-Calibration augmentation (rows 4-7) also improves robustness to FOV changes by 5 to 7 points.

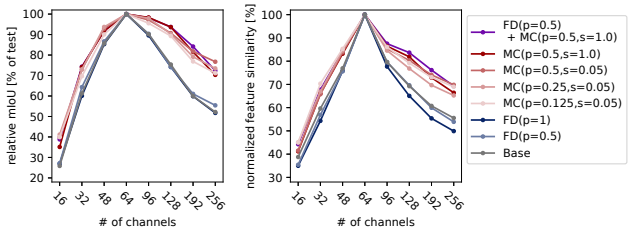
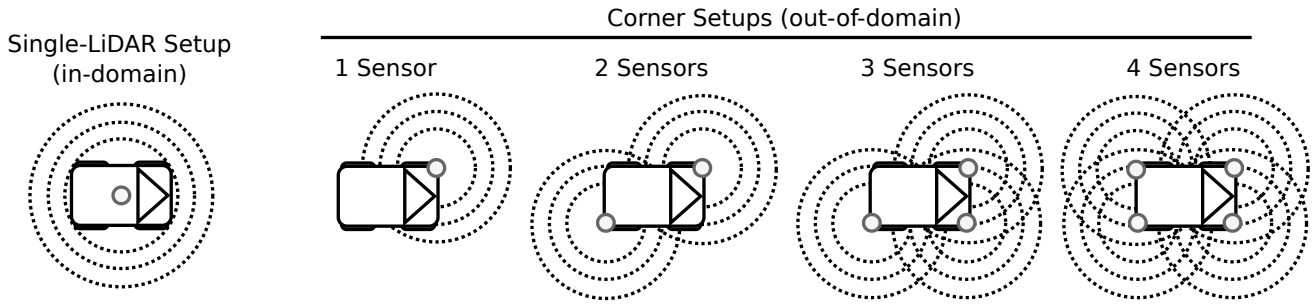


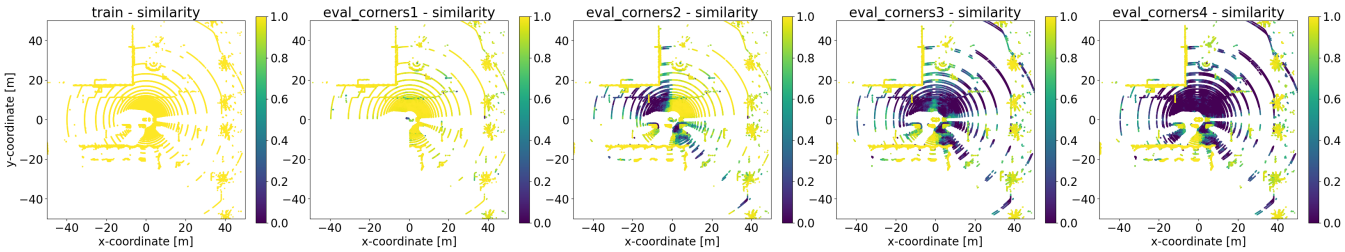
Figure 5: Comparing relative mIoU score (left) and NFS (right) of our proposed augmentations on the test partition of variations of the training setup with varying vertical LiDAR resolution (# of channels). The in-domain setup has 64 channels. mIoU scores are listed in table 1.

Varying the Number of Sensors When using multi-sensor setups, (table 1, columns 3-5) the mIoU of the *Base* model drops steadily from 76.1 to 38.4 points, a 50% drop compared to the original test set mIoU score, also shown in figure 4 (left). This model is unable to generalize to a 2x to 4x increase in the number of points in the input point cloud. Our Frustum Drop augmentation has almost no effect in this evaluation. This is expected,as it typically reduces the number of points in the training data, instead of increasing it. With our strongest Mis-Calibration augmentation (row 7), we can almost entirely mitigate the decrease in mIoU score across the single-sensor and 4-sensor setups. As shown in the last row, both augmentations can be combined to achieve the added robustness of both, at the cost of a comparatively very small drop in in-domain mIoU score.
In figure 4, we compare the relative change in mIoU score (as a percentage of in-domain test set mIoU) to our presented NFS metric. As in figure 2, we again see that our NFS metric closely correlates well with relative mIoU score.

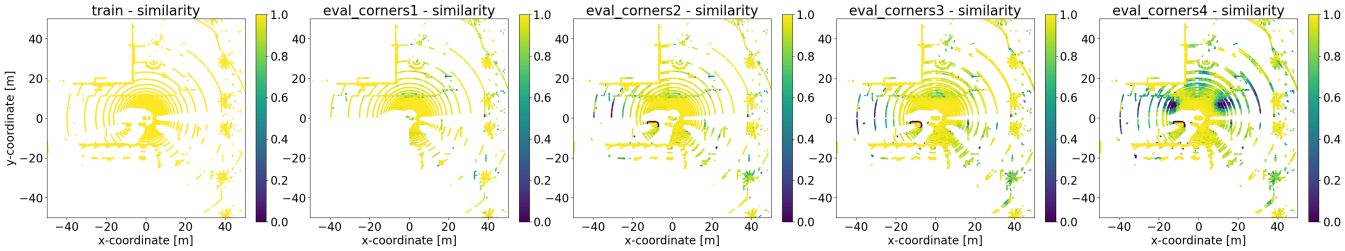
Visualizing Point-wise Feature-Level Similarity In figure 6, we compare the point-wise NFS between the in-domain LiDAR setup and the four out-of-domain *Corner* LiDAR sensor setups, (as used in table 1 left and figure 4). Figure 6(b) shows the normalized feature similarity between in-domain and out-of-domain sensor setups for our baseline model, while the model used in figure 6(c) was trained with both our augmentations. Both models were only trained on the in-domain single-sensor setup (left), and NFS is evaluated across sensor setups on the held-out test split of our simulated sequence. The left column shows the trivial case of comparing in-domain feature vectors to themselves: the NFS is always 100% when comparing the same feature vectors. In the second column, we compare the point-wise features ($\mathbf{f'}_i$) of the single-sensor *Corner* setup to the point-wise features $\mathbf{f}_i$ of the in-domain setup (left). Here, we see that both models are mostly invariant to transitions from one single-sensor setup to another, as shown by an overall high similarity.



(a) A bird's eye view illustration of the different LiDAR setups we simulate in our evaluation.



(b) Without our proposed augmentations, a trained model's features vary strongly between sensor setups.



(c) With our proposed augmentations, the model's features are much more invariant to varying sensor setups.

Figure 6: Features of two Models trained on a single-sensor setup (left), compared across different simulated sensor setups shown in (a). Higher similarity to in-domain features strongly correlates with better generalization.
(b)/(c): Our proposed normalized feature similarity (NFS) applied point-wise between the training setup (left) and the shown setup. Each row shows features from the same trained model.

Augmentation	Val mIoU ↑	NFS ↑	NFS ↑ (1 Sensor vs.)		
	SemanticKITTI	Cross-Sensor	2 Sensors	3 Sensors	4 Sensors
Base	61.3	78.6 ± 3.0	78.8 ± 1.7	70.9 ± 3.6	63.8 ± 3.7
Base + FD(p=0.5) (Ours)	61.4	80.2 ± 2.7	80.1 ± 1.3	70.0 ± 3.2	61.9 ± 3.6
Base + MC(p=0.5, s=1.0) (Ours)	61.4	81.2 ± 2.8	85.8 ± 1.2	82.5 ± 1.6	**80.3** ± 1.0
Base + FD(p=0.5) + MC(p=0.5, s=1.0) (Ours)	**61.8**	**81.6** ± 3.0	**86.2** ± 1.0	**82.7** ± 1.5	80.2 ± 1.1

Table 2: Real-world results of our proposed *Frustrum Drop* (FD) and *Mis-Calibration* (MC) augmentations during zero-shot evaluation on the *un-labeled* CoCar-NextGen dataset (Heinrich et al. 2024). Each row shows one model trained on the SemanticKITTI dataset, with its in-domain validation set mIoU in the first column. We report NFS values on CoCar-NextGen as mean ± std for each setup (X'), using each of the four OS1 sensors as a reference (X in figure 1(b)).

For sensor setups with two and more sensors (figure 6, center and right), we can see that the baseline model (b) is very sensitive to overlapping scans from multiple sensors, showing low feature similarity in overlap regions. The model trained with our augmentations (c) is much more robust to these changes, showing very high feature similarity with up to three sensors, and only starting to be impacted by regions with extremely high point density (right).

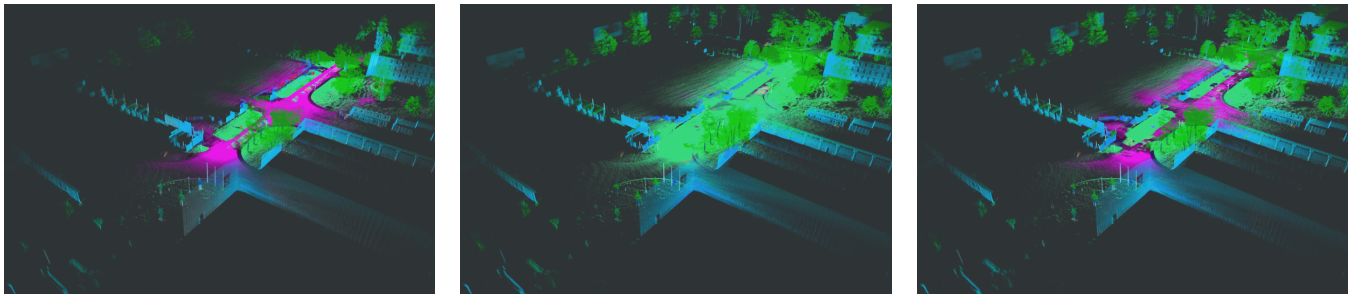
Varying the Sensor Resolution We also compare the robustness of our augmented models by simulating various different sensor resolutions. The results of this are shown in table 1 (right), which lists mIoU scores of the same models applied on variations of a single-sensor setup with a varying number of LiDAR channels. Figure 5 shows the corresponding relative change in mIoU score, as well as our NFS metric. We observe in these results, that our Mis-Calibration augmentation noticeably widens the range of sensor resolutions which the models can generalize to, especially for higher sensor resolutions. While trained on only 64 channels, our model with the strongest augmentation can operate on twice the in-domain resolution, while only losing less than 5 points of mIoU score (74.9 vs 70.2), whereas the baseline model drops from 76.1 to 57.4 points in this range. To a lesser extent, this increased robustness is also seen for numbers of channels lower than 64, but the difference is smaller. This is to be expected, since our Mis-Calibration augmentation increases the number of points in the point cloud. Our Frustum Drop does not have a noticeable effect in this experiment, but can still safely be combined with our Mis-Calibration augmentation without detrimental effects.

5 Results on Real Multi-LiDAR Vehicle Data

In order to verify the invariance improvements of our approach on a real-world high-resolution multi-LiDAR setup, we evaluate our augmentations on a dataset by Heinrich et al. (2024), who provide LiDAR recordings as well as extrinsic calibration files from the multi-LiDAR sensor setup of their research vehicle CoCar-NextGen.

Training on the SemanticKITTI Dataset Since no labels are available for the CoCar-NextGen dataset, we train our models on the single-sensor SemanticKITTI dataset for 18-class semantic segmentation for this experiment. The in-domain mIoU score of our models on the validation set of SemanticKITTI is listed in the first column of table 2. The mIoU scores reported in table 2 are slightly lower than reported by Tang et al. (2020), since we omit the intensity/reflectivity values of our input point clouds, so that our models can work with LiDAR sensors from different sensor manufacturers. Trained with intensity values, our baseline model achieves a validation set mIoU score of 64.1 on SemanticKITTI, which is in line with Tang et al., but fails to generalize to different LiDAR sensors. With our introduced augmentations, we also observe a very slight but consistent uplift in in-domain validation set mIoU score.

Evaluation on the CoCar-NextGen Dataset In this experiment, we evaluate the cross-sensor-setup consistency of our models trained on SemanticKITTI across different single-LiDAR and multi-LiDAR subsets of the CoCar-NextGen sensor setup. As our NFS metric requires aligned point clouds of the same scene, we time-synchronize the scans from the four perception LiDAR sensors on the corners of the roof of CoCar-NextGen, and use KISS-ICP (Vizzo et al. 2023) to apply ego motion correction to the point clouds. We rely on our introduced Normalized Feature Similarity (NFS) metric for this experiment, as it does not require annotated data, and we have demonstrated it to be a reasonable proxy for out-of-domain generalization mIoU when comparing to an in-domain sensor setup in section 3.2 and section 4.3. We use our NFS metric to compare feature-level similarity between one and up to four of the LiDAR sensors from the CoCar-NextGen vehicle in table 2. As a reference for our using four single-LiDAR setups, each comprised of one of the four 128-channel LiDAR sensor at the corners of the vehicle's roof as an in-domain reference point. In the second column of table 2, we report cross-sensor NFS comparing feature consistency between each of the four different LiDAR sensors being processed individually by our models. Here, we see that our baseline model already has high feature similarity between the different sensors (78.6% NFS), but this is slightly increased by our introduced augmentations (81.6% NFS).
In the right three columns, we report the NFS values when comparing features from single sensors to features of fused point clouds from multiple sensors. As we can observe in the first row of table 2, the baseline model is not invariant to the point cloud density, and it's consistency degrades when more than two sensors are fused together, down to



(a) Segmentation for point clouds for a single sensor (front left). Results from multiple time steps are overlaid for visualization, time steps are processed individually by each model.

(b) When applying the same model without our proposed augmentations on four sensors, the model is overwhelmed by the point density of the fused point cloud, and consistently mis-classifies road points as "terrain".

(c) A model trained with our augmentations shows much higher segmentation quality on highly dense point clouds.

Figure 7: Qualitative semantic segmentation results on the CoCar-NextGen dataset, with models trained on SemanticKITTI.

an NFS of 63.8. We show qualitative results for this finding in figure 7(b), where a clear performance degradation from single-sensor (a) to four-sensor (b) can be observed. Our Frustum Drop augmentation (second row in table 2) shows little difference from the baseline in this evaluation. Since we fuse sensors diagonally for our two-sensor setups, as illustrated in Figure 6(a), the combined field of view of multi-sensor setups rarely exhibits blind spots, for which this augmentation was designed. In contrast, our Mis-Calibration augmentation (third row) is highly effective at increasing our model's robustness to multi-LiDAR setups, with NFS scores above 80% for all examined setups. We confirm this with some qualitative results shown in figure 7(c), where the model's segmentation of fused point clouds from four sensors is shown. While this model is trained on data from a sensor setup which typically captures approximately 64,000 points per scan, it now operates on fused point clouds of up to half a million points without a significant loss in segmentation quality.

6 Limitations

In this work, we aim to explore a representative set of LiDAR setups, varying common key design parameters such as FOV, sensor resolution, and number of sensors in a simulated environment. However, an exhaustive evaluation of *all possible* sensor setups far exceeds our computational resources. Additionally, generalization and invariance properties likely also depend on the used model architecture. We focus on a representative voxel-based CNN architecture in this work, since it can operate on multi-LiDAR point clouds without substantial architecture modifications. Future work should investigate how our findings generalize to other architectures such as range image projections, or transformer-based approaches, as well as adjacent tasks such as 3D object detection.

7 Conclusion

In this work, we show that using the right data augmentation strategy can increase the invariance of LiDAR semantic segmentation models, allowing them to generalize from single-sensor datasets to multi-LiDAR vehicle setups without fine-tuning or any other post-training methods being applied.
We propose two specific augmentations to replicate effects commonly present in multi-LiDAR sensor data, when only labeled training data from a single-LiDAR dataset is available.
We also introduce a new method to quantify feature-level invariance we call *Normalized Feature Similarity*, which we demonstrate in simulations with a wide variety of sensor setups to be a good proxy for generalization performance, and strongly correlates with out-of-domain mIoU scores.
Our experiments demonstrate that our augmentations almost entirely alleviate the performance penalty incurred when training on a single-LiDAR setup and evaluating on multi-sensor setups. We validate this using both simulations and real-world data from a vehicle with multiple high-resolution LiDAR sensors.
A promising avenue for future research is the enforcement of invariance through loss functions based on our proposed normalized feature similarity, which is differentiable without further modifications. We therefore see a clear pathway for self-supervised approaches to learn spatially and temporally consistent LiDAR representations across sensor setups using only feature-level supervision with our proposed NFS metric.

Acknowledgment

This work was supported by the German Federal Ministry of Education and Research (BMBF) within the project *MANNHEIM-CeCaS*, funding number 16ME0818.

References

Ayala, R.; and Mohd, T. K. 2021. Sensors in autonomous vehicles: A survey. *Journal of Autonomous Vehicles and Systems*, 1(3): 031003.

Behley, J.; Garbade, M.; Milioto, A.; Quenzel, J.; Behnke, S.; Stachniss, C.; and Gall, J. 2019. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, 9297–9307.

Botev, A.; Bauer, M.; and De, S. 2022. Regularising for invariance to data augmentation improves supervised learning. *arXiv preprint arXiv:2203.03304*.

Bouchacourt, D.; Ibrahim, M.; and Morcos, A. 2021. Grounding inductive biases in natural images: invariance stems from variations in data. *Advances in Neural Information Processing Systems*, 34: 19566–19579.

Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11621–11631.

Chen, H.; Liu, S.; Chen, W.; Li, H.; and Hill, R. 2021. Equivariant point network for 3d point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 14514–14523.

Cohen, T.; and Welling, M. 2016. Group equivariant convolutional networks. In *International conference on machine learning*, 2990–2999. PMLR.

DeVries, T. 2017. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv preprint arXiv:1708.04552*.

Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; and Koltun, V. 2017. CARLA: An open urban driving simulator. In *Conference on robot learning*, 1–16. PMLR.

Fang, J.; Zhou, D.; Zhao, J.; Wu, C.; Tang, C.; Xu, C.-Z.; and Zhang, L. 2024. LiDAR-CS dataset: LiDAR point cloud dataset with cross-sensors for 3D object detection. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 14822–14829. IEEE.

Heinrich, M.; Zipfl, M.; Uecker, M.; Ochs, S.; Gontscharow, M.; Fleck, T.; Doll, J.; Schörner, P.; Hubschneider, C.; Zofka, M. R.; et al. 2024. CoCar NextGen: a Multi-Purpose Platform for Connected Autonomous Driving Research. *arXiv preprint arXiv:2404.17550*.

Karle, P.; Betz, T.; Bosk, M.; Fent, F.; Gehrke, N.; Geisslinger, M.; Gressenbuch, L.; Hafemann, P.; Huber, S.; Hübner, M.; et al. 2023. EDGAR: An Autonomous Driving Research Platform–From Feature Development to Real-World Application. *arXiv preprint arXiv:2309.15492*.

Kimura, M.; Shimizu, R.; Hirakawa, Y.; Goto, R.; and Saito, Y. 2024. On permutation-invariant neural networks. *arXiv preprint arXiv:2403.17410*.

Kvinge, H.; Emerson, T.; Jorgenson, G.; Vasquez, S.; Doster, T.; and Lew, J. 2022. In what ways are deep neural networks invariant and how should we measure this? *Advances in Neural Information Processing Systems*, 35: 32816–32829.

Li, R.; Li, X.; Heng, P.-A.; and Fu, C.-W. 2020. Pointaugment: an auto-augmentation framework for point cloud classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6378–6387.

Li, Y.; Yosinski, J.; Clune, J.; Lipson, H.; and Hopcroft, J. 2015. Convergent learning: Do different neural networks learn the same representations? *arXiv preprint arXiv:1511.07543*.

Liu, Z.; Tang, H.; Lin, Y.; and Han, S. 2019. Point-voxel cnn for efficient 3d deep learning. *Advances in neural information processing systems*, 32.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Tang, H.; Liu, Z.; Zhao, S.; Lin, Y.; Lin, J.; Wang, H.; and Han, S. 2020. Searching efficient 3d architectures with sparse point-voxel convolution. In *European conference on computer vision*, 685–702. Springer.

Triess, L. T.; Peter, D.; Rist, C. B.; and Zöllner, J. M. 2020. Scan-based semantic segmentation of lidar point clouds: An experimental study. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, 1116–1121. IEEE.

Uecker, M.; Fleck, T.; Pflugfelder, M.; and Zöllner, J. M. 2022. Analyzing deep learning representations of point clouds for real-time in-vehicle lidar perception. *arXiv preprint arXiv:2210.14612*.

Vizzo, I.; Guadagnino, T.; Mersch, B.; Wiesmann, L.; Behley, J.; and Stachniss, C. 2023. Kiss-icp: In defense of point-to-point icp–simple, accurate, and robust registration if done the right way. *IEEE Robotics and Automation Letters*, 8(2): 1029–1036.

Weiler, M.; Hamprecht, F. A.; and Storath, M. 2018. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 849–858.

Xiao, A.; Huang, J.; Guan, D.; Cui, K.; Lu, S.; and Shao, L. 2022. Polarmix: A general data augmentation technique for lidar point clouds. *Advances in Neural Information Processing Systems*, 35: 11035–11048.

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R. R.; and Smola, A. J. 2017. Deep sets. *Advances in neural information processing systems*, 30.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, 16259–16268.

Zhu, X.; Zhou, H.; Wang, T.; Hong, F.; Ma, Y.; Li, W.; Li, H.; and Lin, D. 2021. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9939–9948.