Improving Weather-based OOD Generalisation in Lidar-based Object Detection Models via Adversarial Training

Ben Batten
Imperial College London
b.batten@imperial.ac.uk

Alessio Lomuscio
Imperial College London
a.lomuscio@imperial.ac.uk

Abstract

Deep learning models are shown to be effective in point-cloud based object detection tasks, such as interpreting data from lidar sensors collected in on-the-road environments. Key to the use of deep learning in safety-critical applications, such as perception in driverless cars, are assurances on accuracy in a wide range of conditions. Previously, point cloud-based object detection models have performed poorly in on-the-road environments when subject to adverse weather, such as rain, snow, and fog. This stems largely from a lack of broad ranging data during training; when collected naturally, adverse weather data is under-represented in training sets. Previous works have tackled this problem using costly simulations to augment training sets with synthetic adverse weather data. In this paper, we propose to improve adverse weather performance in on-the-road, lidar-based object detection tasks through simulation-free adversarial training. We perform benchmarks using PointRCNN against a state-of-the-art simulation-based approach and improve adverse weather performance by up to 6%.

1. Introduction

Lidar-based perception systems are increasingly being deployed as part of driverless car systems [11, 15, 23, 24]. Moreover, models based on deep learning have demonstrated proficiency in perception tasks built on lidar data, such as segmentation, bounding box regression, and classification problems [5, 8, 14, 16, 25, 26]. The risk factors associated with driverless cars necessitate strong assurances on the performance of deep learning-based perception systems across a wide range of settings. Central to this goal is improving the performance of perception models in *adverse weather*, such as snow, heavy rain, and fog. Adverse weather is often under-represented in training data due to it's relatively infrequent occurrence compared with clear weather. To mitigate this data imbalance, there exist numerous works that aim to accurately simulate the effects of adverse weather on point clouds produced by lidar scanners [3, 7, 9, 10, 22, 28, 33]. In these works, the simulated, *synthetic* data augments the training data with the effect of improving downstream performance in adverse weather settings.

Improving the performance of lidar-based perception models in adverse weather settings is related to broader literature on improving generalisability to out-of-distribution (OOD) data, which has received much attention across the deep learning research landscape. Improving model generalisation to OOD data has been approached using data augmentation [32], invariant representations [1, 2, 6], regularization [13], causal learning [4], and adversarial training [31].

In this work, we approach the problem of OOD generalisation in lidar-based perception models from an adversarial training perspective. Namely, we investigate how adversarial training can be used to improve the performance of point cloud-based deep learning models under unseen, adverse weather scenarios. Although adversarial attacks and training have been previously explored on point cloud-based models [17, 27, 29, 30], we believe this is the first work to demonstrate the effectiveness of adversarial training for tackling the out-of-distribution problem in adverse weather conditions. We perform experiments using point clouds produced by car-mounted lidar sensors in an on-the-road setting. Specifically, in this paper we make three contributions;

- Inspired by [9], we propose a novel point cloud initialisation to emulate fog-affected lidar point clouds in a *simulation-free* manner.
- We introduce and implement an adversarial training scheme on PointRCNN for improving unseen adverse weather performance. Code is made publicly available on GitHub: https://github.com/benbatten/Adversarial-Training-for-LIDAR-OOD-Weather.
- We benchmark our method against a simulation-based approach from [9].

The rest of this work is organised as follows: In Sec. 2 we introduce the necessary background on simulation of adverse weather effects as well as the mathematical framework for adversarial training. In Sec. 3 we outline our proposed method and highlight the design choices made. In Sec. 4 we discuss experimental results on unseen, adverse weather and benchmark against a state-of-the-art simulation-based approach [9] before concluding in Sec. 5.

Related Works

There are several works aimed at improving the performance of lidar-based based perception systems when exposed to out-of-distribution (OOD) data [3, 7, 9, 10, 12, 22, 28, 33]. Of this work, much of it focuses on the adverse weather setting, where rarer weather such as rain, snow, and fog, hereafter collectively termed 'adverse weather', comprise the OOD data. In [3, 9, 10, 12] the authors develop adverse weather simulations which aim to approximate the effects of adverse weather on lidar sensors through simulation. An accurate simulation enables the authors to augment the dataset with partially-synthetic data resembling adverse weather, by applying a simulation to the available clear weather data. Specifically, in [9] the authors model the attenuation of lidar pulses in foggy environments as an exponential decay. Then, given a clear-weather point cloud, they can apply the simulation to modify points affected by the simulated fog. In [12] the authors develop a simulation for rainy environments by modelling raindrops as perfectly spherical Mei scatterers [20]. In [10] the authors model snowfall by extending the scattering-based approach from [12] to include backscattering.

Related to our work is the field of adversarial robustness which has been previously explored on lidar-based perception systems [17, 27, 29, 30]. Although these works examine adversarial attack methods on point cloud-based models, they do not explore leveraging adversarial attacks to improve performance under weather-based OOD settings.

Also related to our work is the field of formal verification which aims to provide guarantees on a network's output given a set of inputs. While the verification literature covers noise-based verification on lidar-based perception models [18, 19, 21], it has not explored threat models related to adverse weather. Moreover, in [21] the authors consider an L_p ball on the position of points in the lidar point cloud. In [19] the authors verify against geometric threat models.

It has previously been shown that adversarial training (AT) can be utilised to improve performance on OOD baselines [31] however, to the best of our knowledge, AT for weather-based OOD generalisation on point cloud models has not been explored.

2. Background

2.1. Adversarial Training (AT)

We consider a neural network $\mathcal{N} = \{L_i | i = 1, \dots, l\}$ as a set of layers $L_i = (W_i, b_i, g_i(\cdot))$ containing parameter matrices W_i and b_i, termed the weights and bias of the layer, and an element-wise, non-linear activation function $g_i(\cdot)$. We define the output of layer i as $f_i(x_{i-1}) = g_i(W_i x_{i-1} + b_i)$ and define the output of the network as $N(x_0) = f_l(x_{l-1})$ and will write the input vector without a subscript for brevity, $x_0 = x$.

Definition 1 (Adversarial Attack) *Given a Network, $\mathcal{N}$, an input point and it's respective class, (x, y), a target class $y_t \neq y$, and an attack budget, ϵ. We define an adversarial attack, x^*, as*

$$x^* = \arg\min_{x'} (\mathcal{L}(N(x'), y_t) \quad s.t. \quad |x - x^*|_p \leq \epsilon, \quad (1)$$

where $|\cdot|_p$ is a L_p-norm and $\mathcal{L}$ is a differentiable loss function.

In a standard training procedure we aim to minimise the network loss with respect to a dataset, $\mathcal{D} = (x^{(i)}, y^{(i)}), i = 1, \dots, n$, by tuning the network parameters $(W_i, b_i), i = 1, \dots, l$. Differently, in adversarial training, we adaptively update our dataset by performing adversarial attacks on the training samples in $\mathcal{D}$. Concretely, the j-th training step consists of,

$$W^{(j+1)} = W^{(j)} - \alpha \nabla_W \left(\mathcal{L}(\mathcal{N}(x^*), y_t)\right) \quad (2)$$

where α is the learning rate and we take W to represent all the weights of the network. As training progresses and the network parameters update, the adversarial attacks are recomputed accordingly with each epoch. While Eq. (2) shows an update for the weight parameters we compute bias updates similarly. Up to now we have described adversarial attacks and training under a classification setting with an identifiable target class y_t. We can extend the definition to regression trivially by maximising over the regression loss with respect to the correct y value,

$$x^* = \arg\max_{x'} (\mathcal{L}_R(N(x'), y) \quad \text{s.t.} \quad |x - x^*|_p \leq \epsilon, \quad (3)$$

where $\mathcal{L}_R(\cdot)$ is a differentiable regression loss function.

2.2. Lidar-based Fog Simulation

In the rest of this paper we focus on improving the OOD performance of lidar-based perception systems in on-the-road environments and where lidar point clouds affected by foggy weather represent the OOD threat. As mentioned in Sec. 1, state-of-the-art approaches to improving OOD performance on adverse weather scenarios leverage simulation techniques to emulate various weather effects on the

lidar point cloud. We now outline the approach introduced in [9] for simulating the effects of fog.

A lidar system operates by emitting pulses of laser light in a given direction and timing the response power and time, with this information the scanner can compute the distance and direction to the object responsible for the reflected signal. In practice a lidar scanner will rotate while continuously transmitting columns of pulses, enabling it to build a 360° scan. We term the power of a pulse transmitted by the scanner as the *transmitted signal power*, P_T, and the return signal as the *received signal power*, P_R. The authors in [9] adopt a linear model, introduced by [22], of the received signal power,

$$P_R(R) = C_A \int_0^{2R/c} P_T(t) H\left(R - \frac{ct}{2}\right) dt, \tag{4}$$

with c the speed of light, R the distance, and $H(\cdot)$ a function modelling the impulse response given by an object at R. The transmitted signal power is given by a time-based sinusoid which factors in the non-instantaneous transmission time, for more detail see Eq. 2 of [9]. We can factorise $H(R)$ into a contribution from the target, and a contribution from optical effects,

$$H(R) = H_C(R) H_T(R). \tag{5}$$

We term the contributions the optical (H_C) and target (H_T) channels. The authors of [9] show that, in a foggy weather setting, we can further decompose $H_T(R)$ into two sources; one from the true, '*hard*' target, and one from fog-induced backscattering, or, the '*soft*' target

$$\begin{aligned} H_T(R) &= H_T^{\text{soft}}(R) + H_T^{\text{hard}}(R) \\ &= \beta U(R_0 - R) + \beta_0 \delta(R - R_0), \end{aligned} \tag{6}$$

where β is a fog-dependent backscattering constant, β_0 is a constant capturing the reflectivity of the hard target, R_0 is the distance to the hard target, $U(\cdot)$ is the Heaviside step function, and $\delta(\cdot)$ is the Dirac delta function. The delta function ensures we only consider the hard signal at the distance it exists, and the Heaviside function ensures we do not model fog effects from behind the hard target. For the derivation of this function please see Eqs. (9-14) in [9]. Further, we can also decompose the received signal power into an erroneous contribution from fog, and a contribution from the hard target

$$P_{R,\text{fog}}(R) = P_{R,\text{fog}}^{\text{hard}}(R) + P_{R,\text{fog}}^{\text{soft}}(R). \tag{7}$$

With $P_{R,\text{fog}}^{\text{hard}}(R)$ an exponential decay with R,

$$P_{R,\text{fog}}^{\text{hard}}(R) = \exp\left(-2\alpha R_0\right) P_{R,\text{clear}}(R), \tag{8}$$

with α a decay constant related to fog density. Moreover, we can express the return signal power for the soft target (the fog) as

$$\begin{aligned} P_{R,\text{fog}}^{\text{soft}}(R) = C_A P_0 \beta \int_0^{2\tau H} \sin^2\left(\frac{\pi}{2\tau_H} t\right) \cdot \\ \frac{\exp(-2\alpha(R - \frac{ct}{2}))}{(R - \frac{ct}{2})^2} \xi(R - \frac{ct}{2}) U(R_0 - R + \frac{ct}{2}) dt, \end{aligned} \tag{9}$$

where P_0 is the peak power of the laser pulse, τ_H a constant describing the time-width of the pulse, and $\xi(\cdot)$ a function modelling the beam crossover of the non-coaxial pulse transmitter and receiver. We can not solve Eq. (9) analytically and so the authors perform numeric integration on each point in the clear-weather point cloud to obtain $P_{R,\text{fog}}^{\text{soft}}$. Computing $P_{R,\text{fog}}^{\text{hard}}$ is trivial. The final step of the simulation is to check $\max_R P_{R,\text{fog}}^{\text{soft}}(R) < P_{R,\text{fog}}^{\text{hard}}(R_0)$, if this holds then the position of the point is unchanged and its intensity is updated to be $P_{R,\text{fog}}^{\text{hard}}(R_0)$. If the condition does not hold then the point is moved to a distance $R' = \arg\max_R P_{R,\text{fog}}^{\text{soft}}(R)$ and the intensity is set to $P_{R,\text{fog}}^{\text{soft}}(R')$. Computing this for every point in the original point cloud concludes the simulation. Here we have outlined the simulation at a high level and we relate it to our contribution in Sec. 3. For a detailed exposition of the simulation see [9].

3. Adversarial Training for Lidar-based Vision Systems

Our approach to improving weather-based OOD generalisation on lidar-based perception models consists of two steps. First, we perform a cheap input initialisation step, and second, we carry out adversarial training. In this section we outline both, beginning with a discussion of the training sample initialisation, after which, we detail our training algorithm and the design choices made before discussing experimental results in Sec. 4.

3.1. Simulation-driven Initialisation

Drawing on key insights from the fog simulation introduced in [9], we propose to initialise training samples prior to adversarial training such that they echo features of the simulation. Specifically, we aim to cheaply mimic how a subset of points from the clear-weather point cloud are erroneously shifted closer to the lidar scanner when fog is simulated. To understand this effect we refer to Fig. 1 which shows the relative $P_{R,\text{fog}}(R)$ for the soft and hard targets. In this example the soft target, i.e., the fog, produces a stronger received power signal, and, accordingly, R' would be interpreted as the distance to an object. It is important here to make two observations; firstly, that whenever the response from $P_{R,\text{fog}}^{\text{hard}}$ is dominant, the distance will be unchanged at R_0, the same as the original distance in the clean, clear-weather point cloud. However, when, as in Fig. 1, the signal from $P_{R,\text{fog}}^{\text{soft}}$ is dominant the point will be shifted to the

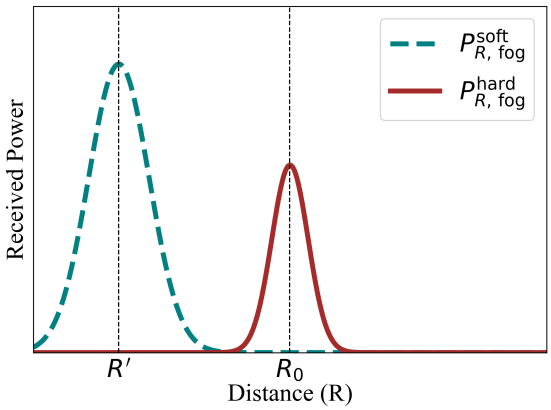


Figure 1. Relative received powers from an erroneous 'soft' target and the true 'hard' target. In this example, backscattering from the fog produces a stronger received signal and so the lidar would interpret R' as the true distance to an object.

distance, R'. To further illustrate this, in Fig. 2 we show a comparison of a clear-weather point cloud and the same cloud after running the fog simulation from [9]. The ring of points around the lidar scanner present in Fig. 2b but lacking in Fig. 2a illustrates the effect described above. Importantly, we note that the simulated points are densely packed in a ring around the scanner. This behaviour can be understood by examining Eq. (9), where we note that only the Heaviside step function $U(R_0 - R + ct/2)$ depends on R_0, the Heaviside function by definition will always be evaluated to 1 for any simulated point. Since no other term depends on the original clear-weather received power $P_{R,\text{clear}}$, or the distance R_0, we can say that for every point in the cloud, the value of R which maximises $P^{\text{soft}}_{R,\text{fog}}$ is a constant, $R' = \arg\max_R(P^{\text{soft}}_{R,\text{fog}})$. The simulated points in Fig. 2b do not sit perfectly on the fixed radius R' as a result of noise independently added to R' for each point. The value of R' is determined by the assumed density of fog being simulated, controlled by α, and by the optical properties of the lidar scanner used, handled in $\xi(\cdot)$. Specifically, the axial separation of the transmitter and receiver.

This 'ring' of simulated points is the key insight we leverage in initialising the adversarial training procedure. Specifically, prior to adversarial training we compute scores, $S(\boldsymbol{p})$, for each point, $\boldsymbol{p}_i = (I_i, [p_x, p_y, p_z]^\intercal_i) = (I, \boldsymbol{p}_{i,xyz})$, in the cloud

$$S(\boldsymbol{p}) = \frac{I}{\exp\left(\sqrt{(p_x^2 + p_y^2 = p_z^2)}\right)} = \frac{I}{\exp(||\boldsymbol{p}_{xyz}||)}, \quad (10)$$

where I is the intensity of a point and is linear with P_R. With the scores computed, we select n points with the lowest scores to be repositioned under our initialisation, where s_{thresh} is the threshold score. We refer to the set of point indexes to be repositioned as $\mathcal{P} = \{i | I_i \leq s_{\text{thresh}}, \forall i\}$. We constrain the new locations of the points to be on the straight line segment joining the original location and the lidar scanner. To determine where on the segment to place the points we choose $R'_{\text{ours}} = \arg\max_R(\xi(R))$, which is the distance at which the transmission beam cone and receiver cone first completely overlap. In [22] the authors discuss in detail non-coaxial beam configurations (also called bistatic beam configurations) and define the distance R'_{ours}, referred to as R_2 in [22]. Once we obtain a value for R'_{ours} on a given lidar scanner we are ready to reposition our n points. Echoing the approach in [9], we add noise independently to the R'_{ours} when setting the final position of each $\boldsymbol{p}_i$. We handle intensity by setting the adjust point intensity to the arithmetic mean of *all* points in the cloud. Concretely, the resulting modified points are given by

$$\begin{aligned} \boldsymbol{p'}_i &= (\bar{I}, \boldsymbol{p}'_{i,xyz}) \\ \boldsymbol{p}'_{i,xyz} &= R'_{\text{ours}} \cdot \frac{\boldsymbol{p}_{i,xyz}}{||\boldsymbol{p}_{i,xyz}||} \\ \bar{I} &= \frac{1}{N}\sum_{j=1}^{N} I_j, \end{aligned} \quad (11)$$

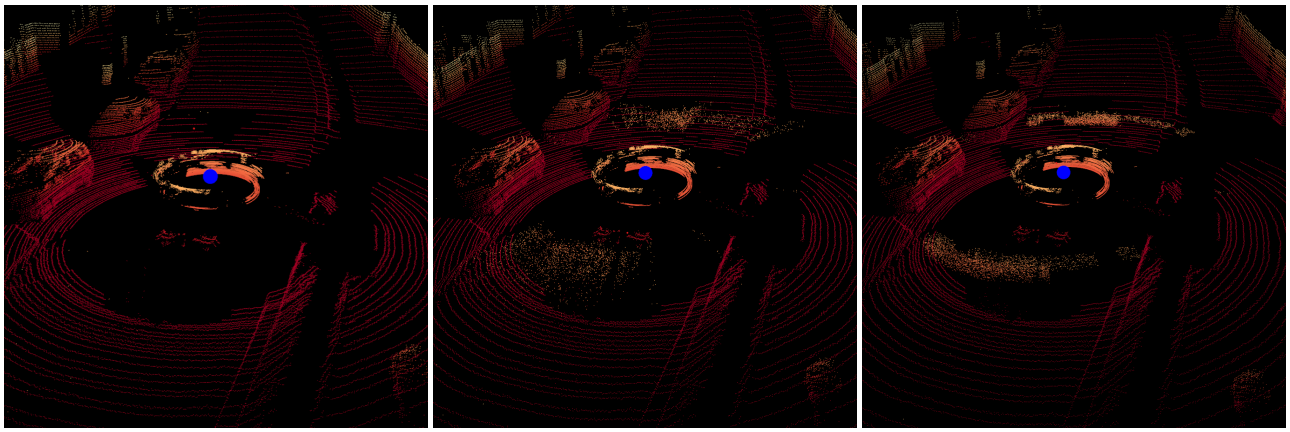
where N is the total number of points in the point cloud. The resulting point cloud is given by

$$\mathcal{C}' = \{\boldsymbol{p}'_i | \forall i \in \mathcal{P}\} \cup \{\boldsymbol{p}_j | \forall j \notin \mathcal{P}\}. \quad (12)$$

This concludes the initialisation step of our procedure. An example of an initialised point cloud is given in Fig. 2c. It's qualitative similarity with the simulation from [9] is perceptible. However, we have produced it without need for computing $P^{\text{soft}}_{R,\text{fog}}$, $P^{\text{hard}}_{R,\text{fog}}$, or R'.

3.2. Adversarial Training on Point Clouds

Once we have initialised a training sample, we update it using an adversarial attack, as described in Definition 1. However, differently to a 2D setting, we have four degrees of freedom for each input point, one intensity value, and three position values. To perform an adversarial attack we modify the notation in Definition 1 to include a budget *vector*, $\boldsymbol{\epsilon} = [\epsilon_I, \epsilon_p]$ which includes attack budgets for the intensity channel and position channels respectively. Moreover, we apply the attack budgets *independently* to each point $\boldsymbol{p}_i$, rather than *jointly* across all the model inputs as is standard in the noise-based attack literature. We note, however, that the L_∞-norm frequently applied jointly across inputs as a budget in noise-based adversarial attacks is equivalent to an input-wise L_1-norm. This is how we approach bounding the intensity channel of each $\boldsymbol{p}_i$. Concretely, $|I_i - I^*_i| \leq \epsilon_I, \forall i \in |\mathcal{C}'|$, where I^*_i is the intensity channel of the i-th point, $\boldsymbol{p}^*_i$, in an adversarial attack, $\mathcal{C}^* = \{\boldsymbol{p}^*_i | \forall i \in |\mathcal{C}'|\}$. For the budget on point position,



(a) Clean (b) Simulated [9] (c) Our Initialisation

Figure 2. Comparison of a clean, clear-weather point cloud, against a point cloud modified by the fog simulation presented in [9]. The lidar scanner is marked by the blue sphere in the centre.

Algorithm 1 Initialisation and Adversarial Training Procedure

Input: Network $\mathcal{N}$, dataset $\mathcal{D} = \{(\boldsymbol{p}^{(i)}, \boldsymbol{y}^{(i)})\}$, number of training steps N, attack budget $\boldsymbol{\epsilon}$, learning rate α, and number of epochs T.

Output: Trained network parameters $\boldsymbol{W}^{(N)}$.

Initialise network parameters $\boldsymbol{W}^{(0)}$ (e.g., randomly or using heuristics)
for $t = 1$ to T **do** ▷ Loop over epochs
  for each mini-batch $\mathcal{B} \subset \mathcal{D}$ **do** ▷ Iterate over mini-batches
    for each input $\boldsymbol{p}^{(i)} \in \mathcal{B}$ **do**
      Initialise input $\boldsymbol{p}'^{(i)}$ according to Eq. (12)
      Compute adversarial input: $\boldsymbol{p}^{*,i} = \arg\max_{\boldsymbol{p}'^{(i)}} \mathcal{L}(\mathcal{N}(\boldsymbol{p}'^{(i)}), \boldsymbol{y}^{(i)})$ s.t.$\|\boldsymbol{p}'^{(i)} - \boldsymbol{p}^{(i)}\| \leq \boldsymbol{\epsilon}$
    end for
    Update network parameters:

$$\boldsymbol{W}^{(n+1)} \leftarrow \boldsymbol{W}^{(n)} - \alpha \nabla_{\boldsymbol{W}} \mathcal{L}(\mathcal{N}(\boldsymbol{p}^*), \boldsymbol{y})$$

  end for
end for
return $\boldsymbol{W}^{(N)}$

$\boldsymbol{\epsilon}_p$, we apply the L_∞ norm element-wise, such that the adjusted points are locally bound to their initial position. This is essential to preserving geometric locality across the point cloud. Concretely, $|\boldsymbol{p}_{i,xyz} - \boldsymbol{p}^*_{i,xyz}|_\infty \leq \epsilon_p |\boldsymbol{p}_{xyz}|, \forall i \in |\mathcal{C}'|$, where $|\boldsymbol{p}_{i,xyz}|$ is a vector of the absolute coordinates x, y, and z for point i. We provide an outline of our training procedure in Algorithm 1.

4. Evaluation

For our experiments we use the lidar portion of the Seeing Through Fog (STF) dataset [3] which consists of 3,469 clear-weather lidar scans for training as well as 1,847 for testing. Importantly, it also provides 1,732 scans collected in foggy conditions. We use PointRCNN [25] as our baseline model. All experiments were run on Ubuntu 22.04 running kernel 6.8.0-49-generic, using an AMD Ryzen Threadripper 3960X with 24 cores, an RTX 3080 Ti and 256GB of RAM.

4.1. Model Training

The PointRCNN [25] model is trained to predict a bounding box in 3D space around each in-class object in the point cloud, and a classification for the bounding box. The STF dataset [3] contains three classes: car, cyclist, and pedestrian. PointRCNN consists of two stages; a ground-up bounding box generation stage, and a bounding box refinement stage. The bounding box generation stage takes a raw point cloud as an input and segments the points into 'foreground' points, which border an in-class object, and 'background' points, which do not. It then proposes a bounding box location for *each* foreground point, this results in multiple bounding box proposals for an object (one for each foreground point lining the object). The bounding box refinement stage takes the bounding box proposals and the foreground mask, along with the original point cloud, and refines the bounding box predictions so there is one bounding box prediction per in-class object. It also classifies the box into one of the target classes. For a detailed breakdown of the architecture see [25].

The attacks in our adversarial training procedure focus only on the first component of the model, the foreground-background segmentation and bounding box proposal gen-

Table 1. Numeric results showing R40 mAP scores across each class and difficulty. *Clean* indicates scores obtained on clear-weather data only. *Fog* indicates score obtained on the foggy test set from STF [25]. The intersection over union (IoU) thresholds are 0.5, 0.25, and 0.25 for the car, cyclist, and pedestrian classes respectively.

Training Method	Car						Cyclist						Pedestrian					
	Easy		Medium		Hard		Easy		Medium		Hard		Easy		Medium		Hard	
	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog
Standard Training	68.9	39.5	69.4	40.0	65.0	38.3	44.2	21.0	43.7	21.0	41.9	22.7	35.4	19.5	32.8	17.5	31.0	16.0
Simulated [9]	29.5	32.3	30.8	31.9	27.1	29.6	16.6	10.1	16.6	10.1	14.9	10.1	5.58	10.6	3.87	9.96	3.96	8.42
Adversarial Training (Ours)	66.0	35.7	68.1	36.3	62.8	35.0	43.8	**24.8**	42.7	**24.8**	40.8	**25.2**	33.3	27.5	32.0	26.0	28.9	22.5
Clean + Simulated [9]	71.9	43.1	69.7	43.3	63.1	41.5	35.1	18.9	35.7	18.9	33.7	19.5	25.8	26.2	23.8	24.2	21.7	22.6
Clean + AT (Ours)	72.2	**45.0**	72.8	**45.1**	67.4	**43.3**	39.6	19.9	39.5	19.9	37.7	20.5	33.3	**32.4**	31.7	**30.4**	28.1	**28.1**

eration. We make this choice as these procedures have differentiable loss. Specifically, PointRCNN uses focal loss when training the point segmentation block, and cross-entropy loss in training the bounding box proposals. Our attacks take place in the raw point cloud input space, as described in Sec. 3.

For each run we train for 50 epochs using either adversarial training, simulated data (collectively called *synthetic* data), clean data, or, where indicated, an augmentation of the above.

4.2. Adversarial Training for Adverse Weather

In Sec. 4 we outline the numeric results of our comparison. The results for our method presented in Sec. 4 all use a position budget $\epsilon_p = 0.001$ and an intensity budget $\epsilon_I = 0$. After experimenting with attacks on both position and intensity we find that fixing the intensity budget to 0 yields the best performance. We propose this is due to the model struggling to converge when the attack space is too large and that for larger numbers of training epoch a combination of position and intensity attacks may further improve performance. Further, we posit that the relative importance of position over intensity could be due to the model focusing more on point position than intensity when predicting bounding boxes. This makes intuitive sense as different surfaces will have different reflectivities, and the same surfaces may return different received powers when at different angles to the sensor. These effects may lead to poor performance when intensity is over-relied on by the model.

We observe that, when using the synthetic data only, our approach achieves over double the clean accuracy of the approach in [9], as well as higher scores on the fog test set. Interestingly, standard training beats both approaches in each setting when the synthetic data is not augmented with clean data during training. A noteworthy feature of the results is that the models trained on simulated data perform better in the fog test set than the clear-weather test set for the car and pedestrian classes. Differently, when trained using adversarial training, the models achieve comparable clean accuracy to the standard training. We believe the adaptive nature of adversarial training combined with the fact that we attack across *all* points in the point cloud prevents our approach from over-fitting to the foggy data in the same way the simulated data does. Moreover, it is well-known that adversarial training comes at a cost to clean performance, however, by using very limited attack budgets this effect is minimised, as shown by the comparable results between AT and Standard Training.

When we train on a combination of clean and synthetic data we see performance improvements for both clear-weather and the fog test sets. In some cases, with Clean + AT achieving the highest clean and fog accuracies simultaneously. A notable exception is for the cyclist class, where clean accuracy decreases for AT when adding clean data. This is a counter-intuitive result. A likely explanation lies in the competing loss objectives of AT and standard training. Moreover, while experimenting with adversarial pre-training followed by standard training, we observe a sudden drop in clean and fog performance when the model is first trained with the clean data. This drop gradually recovers to yield the results shown in Sec. 4. This behaviour eludes to a well-known challenge in adversarial training, that performance is highly sensitive to training schedules. We leave an in-depth study on the impact of different training schedules to further work.

The results for the clean + simulated-trained models are significantly improved over the models trained with simulated data alone. The improvement in clean accuracy is particularly prominent. Interestingly, the fog accuracy has improved across the board, suggesting that training on the simulated data alone is insufficient to learn good representations of the in-class objects.

4.3. Ablations

4.3.1. Attack Budgets

In Fig. 3 we show an ablation of the mAP per class against the position budget ϵ_p used in training. We observe two main features; firstly that the cyclist and pedestrian classes

Table 2. Numeric results showing the relative performance of adversarial training (AT), simulated data [9], and the random initialisation scheme outlined in Sec. 3. Columns report mAP40 average results for the same thresholds presented in Sec. 4.

Training Method	Car						Cyclist						Pedestrian					
	Easy		Medium		Hard		Easy		Medium		Hard		Easy		Medium		Hard	
	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog	Clean	Fog
Standard Training	68.9	39.5	69.4	40.0	65.0	38.3	44.2	21.0	43.7	21.0	41.9	22.7	35.4	19.5	32.8	17.5	31.0	16.0
AT on Simulated Data [9]	17.9	19.9	22.2	19.7	21.8	18.6	11.8	7.96	11.8	8.0	11.8	8.0	4.16	10.0	4.31	8.62	4.28	8.79
Random Initialisation Only	69.7	35.7	70.4	36.6	65.1	35.2	37.2	16.6	37.0	16.6	36.6	17.2	27.2	18.2	26.2	17.4	25.2	15.4
AT Only	66.9	33.4	67.0	34.3	61.7	32.8	40.7	15.0	40.7	15.0	40.5	15.0	38.8	17.6	32.9	26.4	32.4	25.0

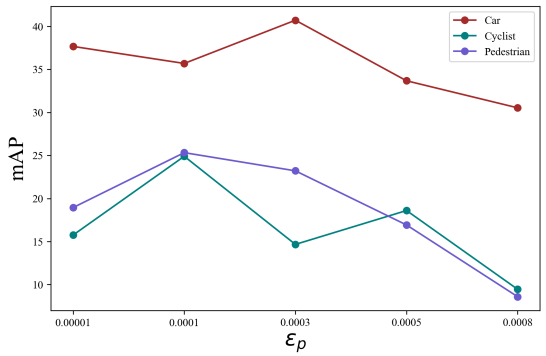


Figure 3. Mean Average Precision (mAP) computed using R40 scores for each class plotted against position budget ϵ_p. Results are for 50 epochs of adversarial training.

appear more sensitive to the budget scale than the car class, and secondly, that the peak mAP for the pedestrian and cyclist classes appears at a lower attack budget than for the car class. It is important to recall that when performing adversarial training we do not limit the attacked to only points moved under the initialisation scheme. We suggest that the relative sensitivities of classes to attack budgets arises from the relative sizes of the objects in the point clouds; namely, a car is bigger than a cyclist or a pedestrian. Moreover, the point clouds are generated using a car-mounted lidar scanner, as a result, other cars are more likely to be close to the scanner than pedestrians, further enhancing the effect. We suggest this is responsible for both of the observations mentioned above.

4.3.2. Decomposing training components

In this ablation, we examine the independent contributions from the different training paradigms studied. Specifically, we seek to identify the unique importance of the adversarial training, random initialisation, and simulated data [9].

In Sec. 4.2 we report numeric results on a combination of the aforementioned training schemes. The numeric results reported here are on the same metrics reported in Sec. 4.2 and we again include the results for standard training to aid comparison. We begin by examining the Random Initialisation Only row, which reports results obtained on models trained using only the initialisation scheme presented in Sec. 3, without any adversarial training. Interestingly, these results are relatively strong, outperforming both the AT on Simulated data and the Pure AT rows. In same cases, they outperform the results for the simulated-only data from Sec. 4. This indicates that our initialisation scheme is effective out of the box, unsurprising given that it is based on the mathematical formulation for the simulation in [9].

When we look at AT Only, we see the performance closely mirrors that for Random Initialisation Only, with the main divergence coming for the smaller object classes, namely Pedestrian. For this class we see much greater performance from AT Only than Random Initialisation Only. This indicates that these two schemes are each contributing in different ways to the overall performance we see for the combined training scheme in Sec. 4. This highlights the importance of using both schemes jointly to achieve the best results.

One of the most interesting and obvious results is for the Simulated Data row, in which we see a complete breakdown in model learning, achieving the worst results across all benchmarks. We attribute this to the challenge of the learning task being too great for the model to learn in the number of epochs used. Specifically, the simulated data is produced by considering the effect of fog on every point in the cloud. Meanwhile, our initialisation scheme only modifies a small fraction of points, and does not attenuate the intensity of all points, as the simulation does. When we combine the starkly different data produced by the simulation with the dynamic learning objective of adversarial training the model appears to fail to capture the data distribution, in particular failing to identify objects in clear weather, as it has not been exposed to these examples in training.

5. Conclusions

In this paper we have introduced a novel initialisation for adversarial training on lidar-produced point cloud data. Further, we propose a method based on our initialisation

and adversarial training to train point cloud-based perception models with the goal of improving performance on OOD data, here represented by adverse weather data. We have shown our proposed method improves OOD performance and have benchmarked it against a state-of-the-art simulation-based approach, observing an accuracy improvement of up to 6% on the simulation-based benchmark. Though we have demonstrated the potential of adversarial training for use in OOD generalisation for point cloud-based perception models, we would like to further explore the effects of training schedules - such as pre-training, integrated training, and fine tuning - on the results we observe; we leave this for future work. Moreover, we leave a more detailed study on the effects of attack budgets and initialisation parameters on empirical performance to future work.

Acknowledgements

Ben Batten was funded by the UKRI Centre for Doctoral Training in Safe and Trusted Artificial Intelligence. Alessio Lomuscio was supported by a Royal Academy of Engineering Chair in Emerging Technologies.

References

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2020. 1

[2] Haoyue Bai, Rui Sun, Lanqing Hong, Fengwei Zhou, Nanyang Ye, Han-Jia Ye, S.-H Chan, and Zhenguo Li. Decaug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35: 6705–6713, 2021. 1

[3] Mario Bijelic, Tobias Gruber, Fahim Mannan, Florian Kraus, Werner Ritter, Klaus C. J. Dietmayer, and Felix Heide. Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11679–11689, 2019. 1, 2, 5

[4] Peter Buhlmann. Invariance, causality and robustness. *Statistical Science*, 2018. 1

[5] R. Qi Charles, Hao Su, Mo Kaichun, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation . In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, Los Alamitos, CA, USA, 2017. IEEE Computer Society. 1

[6] Yongqiang Chen, Wei Huang, Kaiwen Zhou, Yatao Bian, Bo Han, and James Cheng. Understanding and improving feature learning for out-of-distribution generalization. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2024. Curran Associates Inc. 1

[7] Christopher Goodin, Daniel Carruth, Matthew Doude, and Christopher Hudson. Predicting the influence of rain on lidar in adas. *Electronics*, 8:89, 2019. 1, 2

[8] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12):4338–4364, 2021. 1

[9] Martin Hahner, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Fog simulation on real lidar point clouds for 3d object detection in adverse weather. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15263–15272, 2021. 1, 2, 3, 4, 5, 6, 7

[10] Martin Hahner, Christos Sakaridis, Mario Bijelic, Felix Heide, Fisher Yu, Dengxin Dai, and Luc Van Gool. Lidar snowfall simulation for robust 3d object detection. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16343–16353, 2022. 1, 2

[11] Jeff Hecht. Lidar for self-driving cars. *Optics & Photonics News*, 29:26–33, 2018. 1

[12] Velat Kilic, Deepti Hegde, Vishwanath A. Sindagi, A. Brinton Cooper, Mark A. Foster, and Vishal M. Patel. Lidar light scattering augmentation (lisa): Physics-based simulation of adverse weather conditions for 3d object detection. *ArXiv*, abs/2107.07004, 2021. 2

[13] David Krueger, Ethan Caballero, Jörn-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Rémi Le Priol, and Aaron C. Courville. Out-of-distribution generalization via risk extrapolation (rex). In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pages 5815–5826. PMLR, 2021. 1

[14] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12689–12697, 2019. 1

[15] You Li and Javier Ibanez-Guzman. Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems. *IEEE Signal Processing Magazine*, 37(4):50–61, 2020. 1

[16] Ying Li, Lingfei Ma, Zilong Zhong, Fei Liu, Dongpu Cao, Jonathan Li, and Michael A. Chapman. Deep learning for lidar point clouds in autonomous driving: A review, 2020. 1

[17] Daniel Liu, Ronald Yu, and Hao Su. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2279–2283, 2019. 1, 2

[18] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6182–6191, 2021. 2

[19] Tobias Lorenz, Anian Ruoss, Mislav Balunovic, Gagandeep Singh, and Martin Vechev. Robustness Certification for Point Cloud Models . In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7588–7598, Los Alamitos, CA, USA, 2021. IEEE Computer Society. 2

[20] Gustav Mie. Beiträge zur optik trüber medien, speziell kolloidaler metallösungen. *Annalen der Physik*, 330(3):377–445, 1908. 2

[21] Ronghui Mu, Wenjie Ruan, Leandro S. Marcolino, and Qiang Ni. 3dverifier: efficient robustness verification for 3d point cloud models. *Mach. Learn.*, 113(4):1771–1798, 2022. 2

[22] R. Rasshofer, M. Spies, and H. Spies. Influences of weather phenomena on automotive laser radar systems. *Advances in Radio Science*, 9, 2011. 1, 2, 3, 4

[23] Ricardo Roriz, Jorge Cabral, and Tiago Gomes. Automotive lidar technology: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):6282–6297, 2022. 1

[24] Santiago Royo and Maria Ballesta-Garcia. An overview of lidar imaging systems for autonomous vehicles. *Applied Sciences*, 9(19), 2019. 1

[25] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–779, 2019. 1, 5, 6

[26] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10526–10535, 2020. 1

[27] Jiachen Sun, Karl Koenig, Yulong Cao, Qi Alfred Chen, and Zhuoqing Morley Mao. On adversarial robustness of 3d point cloud classification under adaptive attacks. In *British Machine Vision Conference*, 2021. 1, 2

[28] Robin Wang, Yibo Yang, and Dacheng Tao. Art-point: Improving rotation robustness of point cloud classifiers via adversarial rotation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14351–14360, 2022. 1, 2

[29] Chong Xiang, C. Qi, and Bo Li. Generating 3d adversarial point clouds. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9128–9136, 2018. 1, 2

[30] Jiancheng Yang, Qiang Zhang, Rongyao Fang, Bingbing Ni, Jinxian Liu, and Qi Tian. Adversarial attack and defense on point sets. *ArXiv*, abs/1902.10899, 2019. 1, 2

[31] Mingyang Yi, Lu Hou, Jiacheng Sun, Lifeng Shang, Xin Jiang, Qun Liu, and Zhiming Ma. Improved ood generalization via adversarial training and pretraing. In *Proceedings of the 38th International Conference on Machine Learning*, pages 11987–11997. PMLR, 2021. 1, 2

[32] Hongyi Zhang, Moustapha Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ArXiv*, abs/1710.09412, 2017. 1

[33] Yue Zhao, Yuwei Wu, Caihua Chen, and Andrew Lim. On isometry robustness of deep 3d point cloud models under adversarial attacks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1198–1207, 2020. 1, 2