Accurate 3D Object Detection using Energy-Based Models

Fredrik K. Gustafsson[1] Martin Danelljan[2] Thomas B. Schön[1]

[1]Department of Information Technology, Uppsala University, Sweden
[2]Computer Vision Lab, ETH Zürich, Switzerland

Abstract

Accurate 3D object detection (3DOD) is crucial for safe navigation of complex environments by autonomous robots. Regressing accurate 3D bounding boxes in cluttered environments based on sparse LiDAR data is however a highly challenging problem. We address this task by exploring recent advances in conditional energy-based models (EBMs) for probabilistic regression. While methods employing EBMs for regression have demonstrated impressive performance on 2D object detection in images, these techniques are not directly applicable to 3D bounding boxes. In this work, we therefore design a differentiable pooling operator for 3D bounding boxes, serving as the core module of our EBM network. We further integrate this general approach into the state-of-the-art 3D object detector SA-SSD. On the KITTI dataset, our proposed approach consistently outperforms the SA-SSD baseline across all 3DOD metrics, demonstrating the potential of EBM-based regression for highly accurate 3DOD. Code is available at https://github.com/fregu856/ebms_3dod.

1. Introduction

3D object detection (3DOD) is a key perception task for self-driving vehicles and other autonomous robots. 3DOD entails detecting various objects from sensor data, and estimating their size and position in the 3D world. Specifically, the goal of 3DOD is to place oriented 3D bounding boxes which tightly contain all surrounding objects of interest. See Figure 1 for an example. These 3D bounding boxes then serve as input to important high-level tasks such as planning and collision avoidance. Accurate 3DOD is thus crucial for safe autonomous navigation of different complex environments.

In the automotive domain, 3DOD is usually performed from LiDAR point clouds [65, 56, 54], images captured by vehicle-mounted cameras [59, 5, 58], or from a combination of both data modalities [36, 37, 32]. Radar sensors are sometimes also utilized [42, 43, 63]. State-of-the-art 3D object detectors employ deep neural networks (DNNs) to learn powerful feature representations directly from this data [54, 51, 66]. The 3DOD task is then commonly divided into two sub-tasks, in which anchor or proposal 3D bounding boxes are classified as either background or a specific class of object, and then regressed toward ground truth boxes [68, 33, 55].

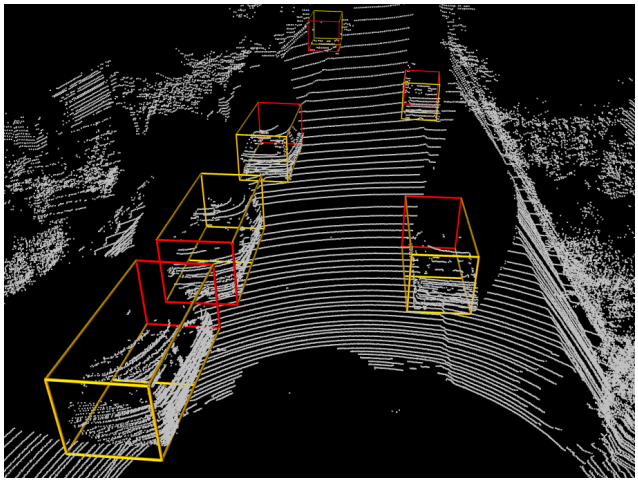

Figure 1. We study how energy-based models (EBMs) can be applied to accurately regress 3D bounding boxes in 3DOD from LiDAR point clouds. Here, we visualize the output of our detector on a validation example from the KITTI [15] dataset.

In general, regression entails predicting a continuous target y from an input x. This is a fundamental machine learning problem that can be addressed using a variety of different techniques [34, 14, 40, 49, 7]. Specifically in 3DOD, the 3D bounding box regression problem is usually addressed by letting a DNN directly predict a target bounding box y for a given input x, and training the DNN by minimizing the L^2 or Huber loss [28, 68, 65, 54, 23]. Alternatively, a probabilistic regression approach has also been employed. The conditional target density $p(y|x)$, i.e. the distribution for the target 3D bounding box y given the input x, is then explicitly modelled using a DNN, which is trained by minimizing the associated negative log-likelihood. Previous

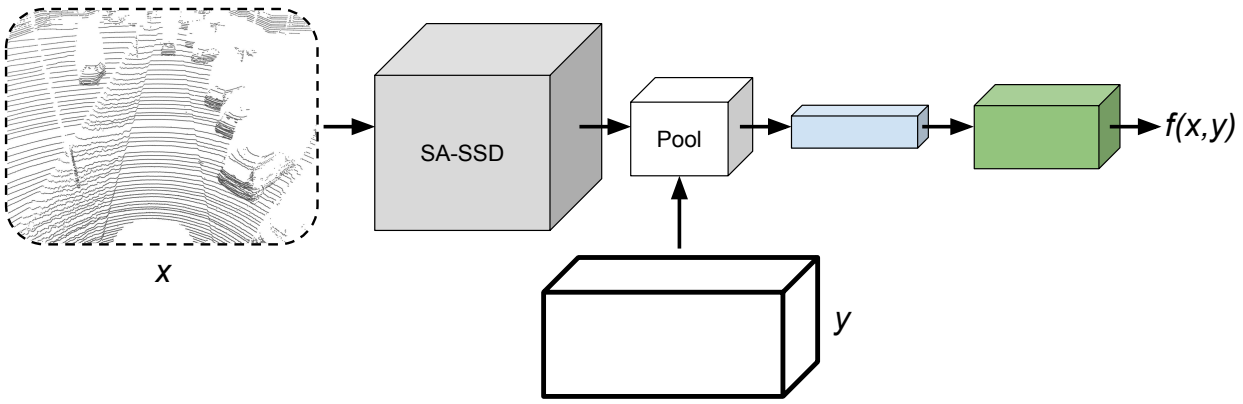

Figure 2. An overview of our proposed approach, applying EBM-based regression to the task of 3D object detection. We integrate a conditional EBM $p(y|x;\theta) = e^{f_\theta(x,y)} / \int e^{f_\theta(x,\tilde{y})} d\tilde{y}$ into the state-of-the-art 3D object detector SA-SSD [23]. We achieve this by designing a differentiable pooling operator that, given a 3D bounding box y, extracts a feature vector from the SA-SSD output. This feature vector is then processed by three fully-connected layers, outputting the scalar energy $f_\theta(x, y) \in \mathbb{R}$.

work on 3DOD has mainly explored Gaussian models of $p(y|x)$ [9, 11, 10, 41].

A Gaussian model is however fairly restrictive, limiting $p(y|x)$ to unimodal and symmetric distributions. Instead, recent work [19, 6, 20] has demonstrated that improved regression accuracy can be obtained on various tasks by employing energy-based models (EBMs) [35] to represent the conditional target density $p(y|x)$. Specifically, this approach entails modeling $p(y|x)$ with the conditional EBM $p(y|x;\theta) = e^{f_\theta(x,y)} / \int e^{f_\theta(x,\tilde{y})} d\tilde{y}$, and then using gradient ascent to maximize $p(y|x;\theta)$ w.r.t. y at test-time. Since the EBM $p(y|x;\theta)$ is directly specified via the scalar function $f_\theta(x, y)$, which is defined using a DNN, it is a highly expressive model that puts minimal restricting assumptions on $p(y|x)$. Even potential multi-modality in the distribution $p(y|x)$ can therefore be learned directly from data. This EBM-based regression approach is thus an attractive alternative also for 3D bounding box regression, especially considering the impressive performance demonstrated on conventional 2D bounding box regression in images [19, 6, 20].

Extending the approach from 2D to 3D is however challenging. In particular, using gradient ascent to maximize the EBM $p(y|x;\theta)$ at test-time requires the scalar DNN output $f_\theta(x, y)$ to be differentiable w.r.t. the bounding box y. For 2D bounding boxes in images, this was achieved by applying a differentiable pooling operator [30] on image features [19, 6, 20], but this technique is not directly applicable to 3D bounding boxes. How EBM-based regression should be applied to 3DOD is thus currently an open question, which we set out to investigate in this work.

Contributions We apply conditional EBMs $p(y|x;\theta)$ to the task of 3D bounding box regression, extending the recent EBM-based regression approach [19, 6, 20] from 2D to 3D object detection. This is achieved by adding an extra network branch to the state-of-the-art 3D object detector SA-SSD [23], and designing a differentiable pooling operator for 3D bounding boxes y. We evaluate our proposed detector on the KITTI [15] dataset and consistently outperform the SA-SSD baseline detector across all 3DOD metrics. Our work thus demonstrates the potential of EBM-based regression for highly accurate 3DOD.

2. Energy-Based Models for Regression

EBMs were extensively studied by the machine learning community in the past [35, 60, 3, 45, 25, 48]. In recent years they have also had a resurgence within the field of computer vision, frequently being employed for generative image modeling [61, 12, 47, 8, 18, 13, 50, 2]. In comparison, the application of EBMs to regression problems has not been a particularly well-studied topic. Very recent work [19, 6, 20] has however demonstrated their efficacy on diverse computer vision regression tasks such as visual object tracking, head-pose estimation and age estimation.

In regression, the task is to learn to predict targets $y^\star \in \mathcal{Y}$ from inputs $x^\star \in \mathcal{X}$, given a training set $\mathcal{D}$ of i.i.d. input-target pairs, $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, $(x_i, y_i) \sim p(x, y)$. The input space $\mathcal{X}$ depends on the specific problem, but can e.g. correspond to the space of images or point clouds. The target space $\mathcal{Y}$ is continuous, $\mathcal{Y} = \mathbb{R}^K$ for some $K \geq 1$.

In EBM-based regression [19, 6, 20], this task is addressed by modelling the distribution $p(y|x)$ of y given x with a conditional EBM $p(y|x;\theta)$, defined according to,

$$p(y|x;\theta) = \frac{e^{f_\theta(x,y)}}{Z(x,\theta)}, \qquad Z(x,\theta) = \int e^{f_\theta(x,\tilde{y})} d\tilde{y}. \quad (1)$$

Here, $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is a DNN that maps any input-target pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$ directly to a scalar $f_\theta(x, y) \in \mathbb{R}$, and $Z(x, \theta)$ is the input-dependent normalizing partition function. The DNN output $f_\theta(x, y)$ is interpreted as the (negative) energy of the distribution $p(y|x;\theta)$.

2.1. Prediction

At test-time, EBM-based regression entails predicting the most likely target under the model given an input $x^\star$, i.e. $y^\star = \arg\max_y p(y|x^\star;\theta) = \arg\max_y f_\theta(x^\star, y)$. In practice, $y^\star = \arg\max_y f_\theta(x^\star, y)$ is approximated by refining an initial estimate $\hat{y}$ via T steps of gradient ascent,

$$y \leftarrow y + \lambda \nabla_y f_\theta(x^\star, y), \tag{2}$$

thus finding a local maximum of $f_\theta(x^\star, y)$. Evaluation of the partition function $Z(x^\star, \theta)$ is therefore not required.

2.2. Training

The DNN $f_\theta(x, y)$ that specifies the conditional EBM (1) can be trained using various methods for fitting a density $p(y|x;\theta)$ to observed data $\{(x_i, y_i)\}_{i=1}^N$. Generally, the most straightforward such method is probably to minimize the negative log-likelihood $\mathcal{L}(\theta) = -\sum_{i=1}^N \log p(y_i|x_i;\theta)$, which for the EBM $p(y|x;\theta)$ is given by,

$$\mathcal{L}(\theta) = \sum_{i=1}^N \log\left(\int e^{f_\theta(x_i,y)} dy\right) - f_\theta(x_i, y_i). \tag{3}$$

The integral in (3) is however intractable, preventing exact evaluation of $\mathcal{L}(\theta)$. One possible solution to this problem is to approximate the intractable integral using importance sampling, as employed in [19]. However, numerous alternative approaches also exist, including noise contrastive estimation (NCE) [21] and score matching [29]. The problem of how EBMs should be trained specifically for regression was studied in detail in [20], comparing six methods on the task of 2D bounding box regression in images. From this comparison, [20] concluded that a simple extension of NCE should be considered the go-to training method.

NCE entails learning to discriminate between observed data examples and samples drawn from a noise distribution. NCE was adopted for EBM-based regression only recently in [20], but has often been used to train EBMs for classification tasks in the past [46, 44, 31, 39]. Recently, it has also become highly utilized within self-supervised representation learning [26, 1, 4, 22]. Applying NCE to regression means training the DNN $f_\theta(x, y)$ by minimizing the loss,

$$\begin{aligned} J(\theta) &= -\frac{1}{N}\sum_{i=1}^N J_i(\theta), \\ J_i(\theta) &= \log \frac{\exp\{f_\theta(x_i, y_i^{(0)}) - \log q(y_i^{(0)}|y_i)\}}{\sum_{m=0}^M \exp\{f_\theta(x_i, y_i^{(m)}) - \log q(y_i^{(m)}|y_i)\}}, \end{aligned} \tag{4}$$

where $y_i^{(0)} \triangleq y_i$, and $\{y_i^{(m)}\}_{m=1}^M$ are M samples drawn from a noise distribution $q(y|y_i)$ that depends on the true target y_i. Effectively, $J(\theta)$ in (4) is the softmax cross-entropy loss for a classification problem with $M+1$ classes. A simple choice for $q(y|y_i)$ that was shown effective in [20] is setting q to a mixture of K Gaussians centered at y_i,

$$q(y|y_i) = \frac{1}{K}\sum_{k=1}^K \mathcal{N}(y; y_i, \sigma_k^2 I), \tag{5}$$

where K and the variances $\{\sigma_k^2\}_{k=1}^K$ are hyperparameters.

A simple extension to NCE, termed NCE+, was proposed and demonstrated to further improve the regression accuracy on certain tasks in [20]. The DNN f_θ is still trained by minimizing $J(\theta)$ in (4), but $y_i^{(0)}$ is now defined as $y_i^{(0)} \triangleq y_i + \nu_i$. The true target y_i is thus perturbed with $\nu_i \sim q_\beta(y)$, where q_β is a zero-centered and scaled version of $q(y|y_i)$ in (5), i.e. $q_\beta(y) = \frac{1}{K}\sum_{k=1}^K \mathcal{N}(y; 0, \beta\sigma_k^2 I)$. NCE+ accounts for possible inaccuracies in the annotation process producing y_i, and can be understood as a direct generalization of NCE. In fact, NCE is recovered as a special case when $\beta \to 0$ in $q_\beta(y)$.

3. Method

We apply EBM-based regression to 3DOD by extending the state-of-the-art 3D object detector SA-SSD [23] with a conditional EBM $p(y|x;\theta)$ (1). In Sec. 3.1, we first provide necessary background on SA-SSD, including a description of its input and output data format. We then detail how the EBM $p(y|x;\theta)$ is defined, employing differentiable pooling of 3D bounding boxes y and an added network branch, in Sec. 3.2. Our approach for training $p(y|x;\theta)$ is based on NCE and further described in Sec. 3.3. Lastly, our prediction strategy using gradient ascent is detailed in Sec. 3.4.

3.1. The SA-SSD 3D Object Detector

SA-SSD [23] takes a LiDAR point cloud of the scene as input x and produces a set $\{d_i\}_{i=1}^D$ of D detections. Each detection d consists of a predicted 3D bounding box y,

$$y = [\,c_x \quad c_y \quad c_z \quad h \quad w \quad l \quad \phi\,]^\mathsf{T} \in \mathbb{R}^7, \tag{6}$$

and an associated classification confidence score $s \in (0, 1)$. In (6), (c_x, c_y, c_z) is the 3D coordinate of the bounding box center, (h, w, l) is the 3D bounding box size, and ϕ is the heading angle of the bounding box.

The input LiDAR point cloud $x = \{(p_x^{(i)}, p_y^{(i)}, p_z^{(i)})\}_{i=1}^n$ of n points is encoded into a sparse 3D tensor by means of voxelization. This tensor is then processed by a backbone network utilizing submanifold sparse 3D convolutional layers [62, 17], producing a 3D feature tensor $h_1(x)$ of shape $W \times L \times H \times C$. A bird's eye view (BEV) feature representation of the scene is then created by flattening $h_1(x)$ into the 2D feature map $h_2(x)$ of shape $W \times L \times HC$. Then, $h_2(x)$ is further processed by six standard 2D convolutional layers, outputting the feature map $h_3(x)$ of shape $W \times L \times C'$.

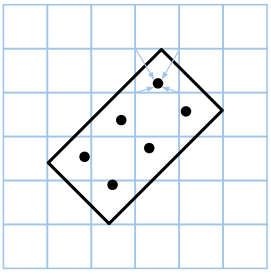

Figure 3. Illustration of our modified variant of RoIAlign [24] for oriented 2D bounding boxes. In this example, the regular $W' \times L'$ grid is 2×3. Bilinear interpolation is used to extract a feature vector for each of the $W'L'$ grid points.

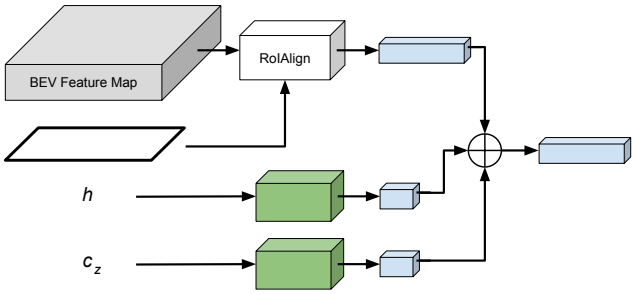

Figure 4. Detailed illustration of the proposed differentiable pooling operation from 3D bounding box y (6) to feature vector $h_5(x, y)$ (8). The BEV version of y is pooled with the BEV feature map produced by SA-SSD. The z coordinate c_z and height h of the box y are processed by two fully-connected layers.

Finally, $h_3(x)$ is fed to a detection network, in which two 1×1 convolutions are applied. The first outputs classification confidence scores and the second outputs offsets for a $W \times L$ grid of anchor 3D bounding boxes.

The SA-SSD backbone and detection networks are trained by minimizing a weighted sum of multiple losses. The focal loss [38] is employed for the classification subtask, and the Huber loss [28] is used for the regression of anchor bounding box offsets. Additionally, SA-SSD employs two losses stemming from auxiliary tasks. By inverting the voxelization via interpolation, 3D feature tensors in the backbone network are represented as point-wise feature vectors. These are then utilized for point-wise foreground segmentation, i.e. predicting whether or not a point lies within any ground truth 3D bounding box, and point-wise center offset regression, i.e. predicting the offset from a foreground point to the center of its 3D bounding box.

3.2. Conditional EBM Definition

In this work, we extend the SA-SSD 3D object detector with a conditional EBM $p(y|x;\theta) = e^{f_\theta(x,y)} / \int e^{f_\theta(x,\tilde{y})} d\tilde{y}$, which is fully specified by the DNN f_θ. To enable the use of gradient ascent at test time (Sec. 2.1), the DNN must be designed such that its scalar output $f_\theta(x, y)$ is differentiable w.r.t. the 3D bounding box y (6). To achieve this, we take inspiration from the recent work [19, 6, 20] applying EBM-based regression to 2D bounding box regression in images. Thus, we design a differentiable pooling operator that, for a given 3D bounding box y, extracts a feature vector from the SA-SSD backbone network output. This feature vector is then processed by an added network branch of fully-connected layers, outputting the energy value $f_\theta(x, y) \in \mathbb{R}$.

Differentiable Pooling of 3D Bounding Boxes Various pooling operators for 3D bounding boxes y (6) have been utilized for refining proposal bounding boxes in previous work [55, 56, 65, 54], none of which are however differentiable w.r.t. the bounding box y. [55] extracts all points in the point cloud $x = \{(p_x^{(i)}, p_y^{(i)}, p_z^{(i)})\}_{i=1}^n$ which lie within a given box y, and then processes the associated point-wise features to extract a feature vector for y. This operator is however not differentiable w.r.t. y, due to the required discrete assessment of whether a point $(p_x^{(i)}, p_y^{(i)}, p_z^{(i)})$ lies within the 3D bounding box y or not. [56] instead divides the box y into a 3D grid and extracts all points which lie within each grid cell. By also encoding which grid cells are empty, this pooling operator better captures geometric information. Because of the discrete extraction of points for each grid cell, it is however still not differentiable w.r.t. the 3D bounding box y. For similar reasons, the pooling operators utilized in [65, 54], which capture even richer contextual information, are not differentiable w.r.t. y either.

Instead, we utilize the 2D feature map $h_3(x)$ of shape $W \times L \times C'$ that is produced by the SA-SSD backbone network. This is a compact yet powerful BEV feature representation of the scene. Specifically, we extract a feature vector $h_4(x, y^{\text{BEV}})$ by pooling $h_3(x)$ with y^{BEV},

$$y^{\text{BEV}} = [\, c_x \quad c_y \quad w \quad l \quad \phi \,]^\mathsf{T} \in \mathbb{R}^5, \tag{7}$$

which is the BEV version of the 3D bounding box y (6). Since y^{BEV} is an oriented 2D bounding box and not necessarily axis-aligned, we can not directly apply standard 2D bounding box pooling operators [16, 24, 30]. Instead we employ a modified variant of RoIAlign [24], which entails dividing y^{BEV} into a regular $W' \times L'$ grid, and extracting a feature vector $g \in \mathbb{R}^{C'}$ in each grid point via bilinear interpolation of $h_3(x)$. See Figure 3 for an illustration. This operation results in a 2D feature map of shape $W' \times L' \times C'$, which we then flatten to obtain the feature vector $h_4(x, y^{\text{BEV}}) \in \mathbb{R}^{W'L'C'}$. By flattening the feature map instead of e.g. averaging over it, more information is preserved in $h_4(x, y^{\text{BEV}})$. It can thus be used to discriminate between a given box and the same box rotated π rad.

This pooling operation is differentiable w.r.t. y^{BEV}, but the extracted feature vector $h_4(x, y^{\text{BEV}}) \in \mathbb{R}^{W'L'C'}$ is of course only a function of y^{BEV} (7), not of the full 3D bounding box y (6). Using gradient ascent at test-time

would thus not update the z coordinate c_z or height h of the bounding box y. To resolve this, we take inspiration from the architecture used for EBM-based age estimation [19]. We thus process $c_z \in \mathbb{R}$ and $h \in \mathbb{R}$ by two small fully-connected layers, generating feature vectors $g_{c_z} \in \mathbb{R}^{C''}$ and $g_h \in \mathbb{R}^{C''}$. Finally, we concatenate the three vectors to obtain $h_5(x, y)$,

$$h_5(x, y) = h_4(x, y^{\text{BEV}}) \oplus g_{c_z} \oplus g_h \in \mathbb{R}^{W'L'C'+2C''}, \quad (8)$$

where $\oplus$ indicates vector concatenation. The complete pooling operation from 3D bounding box y to feature vector $h_5(x, y)$ is illustrated in Figure 4.

Energy Prediction Branch Following [19, 6, 20], we add an extra network branch onto SA-SSD for processing the extracted feature vector. The network branch consists of three fully-connected layers. It takes the feature vector $h_5(x, y) \in \mathbb{R}^{W'L'C'+2C''}$ as input and outputs the scalar energy $f_\theta(x, y) \in \mathbb{R}$, thus fully specifying the conditional EBM $p(y|x; \theta)$ (1). The complete architecture of $p(y|x; \theta)$ is illustrated in Figure 2.

3.3. Detector Training

Following the work on EBM-based 2D object detection [19, 20], the extra fully-connected layers described in Sec 3.2 are added onto a pre-trained and fixed SA-SSD detector. The parameters θ in $f_\theta(x, y)$ thus only stem from these added fully-connected layers, and the SA-SSD backbone and detection networks are kept fixed during training of the DNN f_θ. To train f_θ, we use NCE as described in Sec 2.2. We employ the same training parameters (batch size, data augmentation etc.) as for SA-SSD [23], only replacing the original detector loss with the NCE loss (4).

Algorithm 1 Gradient-based refinement.

Input: $x^\star$, $\{\hat{y}_i\}_{i=1}^D$, T, λ, η.

for $i = 1, \ldots, D$ **do**
  $y \leftarrow \hat{y}_i$.
  for $t = 1, \ldots, T$ **do**
    $\texttt{PrevValue} \leftarrow f_\theta(x^\star, y)$.
    $\tilde{y} \leftarrow y + \lambda \nabla_y f_\theta(x^\star, y)$.
    $\texttt{NewValue} \leftarrow f_\theta(x^\star, \tilde{y})$.
    if $\texttt{NewValue} > \texttt{PrevValue}$ **then**
      $y \leftarrow \tilde{y}$.
    else
      $\lambda \leftarrow \eta\lambda$.
  $y_i \leftarrow y$.
Return $\{y_i\}_{i=1}^D$.

3.4. Detector Inference

At test-time, the input LiDAR point cloud $x^\star$ is first processed by the SA-SSD detector. SA-SSD outputs the 2D feature map $h_3(x^\star)$ and a set $\{(\hat{y}_i, s_i)\}_{i=1}^D$ of D detections, where $\hat{y}_i$ is a 3D bounding box (6) and s_i is the associated classification confidence score. We then take all bounding boxes $\{\hat{y}_i\}_{i=1}^D$ as initial estimates and refine these via T steps of gradient ascent (Sec 2.1), producing $\{y_i\}_{i=1}^D$. The initial 3D bounding boxes $\{\hat{y}_i\}_{i=1}^D$ are thus refined by being moved toward different local maxima of $f_\theta(x^\star, y)$. The refined boxes $\{y_i\}_{i=1}^D$ are finally combined with the original confidence scores, returning the detections $\{(y_i, s_i)\}_{i=1}^D$.

This gradient-based refinement of the detections produced by SA-SSD of course lowers the detector inference speed somewhat. The point cloud $x^\star$ is however still processed by SA-SSD only once, and the scalar $f_\theta(x^\star, y)$ is extracted from $h_3(x^\star)$ using an efficient pooling operator and just a few fully-connected layers. Moreover, the gradient $\nabla_y f_\theta(x^\star, y)$ can be efficiently evaluated using automatic differentiation. The complete refinement procedure is detailed in Algorithm 1, where λ denotes the gradient ascent step-length, η is a decay of the step-length, and the $\texttt{NewValue} > \texttt{PrevValue}$ check ensures that $f_\theta(x^\star, y)$ is never decreased.

	3D @ 0.7			BEV @ 0.7		
	Easy	Moderate	Hard	Easy	Moderate	Hard
Part-A^2 [56]	87.81	78.49	73.51	91.70	87.79	84.61
SERCNN [67]	87.74	78.96	74.30	94.11	88.10	83.43
EPNet [27]	89.81	79.28	74.59	94.22	88.47	83.69
Point-GNN [57]	88.33	79.47	72.29	93.11	89.17	83.90
3DSSD [64]	88.36	79.57	74.55	92.66	89.02	85.86
STD [65]	87.95	79.71	75.09	94.74	89.19	86.42
SA-SSD [23]	88.75	79.79	74.16	95.03	**91.03**	85.96
3D-CVF [66]	89.20	80.05	73.11	93.52	89.56	82.45
CLOCs-PVCas [51]	88.94	80.67	**77.15**	93.05	89.80	**86.57**
PV-RCNN [54]	90.25	**81.43**	76.82	94.98	90.65	86.14
SA-SSD	88.80	79.52	72.30	95.44	89.63	84.34
SA-SSD+EBM	**91.05**	80.12	72.78	**95.64**	89.86	84.56
Rel. Improvement	+2.53%	+0.75%	+0.66%	+0.21%	+0.26%	+0.26%

Table 1. Results on KITTI test in terms of 3D and BEV AP. Our SA-SSD+EBM detector consistently outperforms the SA-SSD baseline, and achieves highly competitive performance also compared to other state-of-the-art methods.

4. Experiments

We evaluate our EBM-based 3DOD approach on the KITTI 3DOD dataset [15] and compare it with the SA-SSD [23] baseline and other state-of-the-art methods. Our detector is implemented in PyTorch [52]. Training and inference code is publicly available.

4.1. Dataset

KITTI [15] is the most commonly used dataset for automotive 3DOD. It contains 7 481 examples for training, and 7 518 *test* examples without publicly available ground truth annotations. Following common practice [23, 54], the training examples are further divided into *train* (3 712 ex-

	3D @ 0.7			BEV @ 0.7		
	Easy	Moderate	Hard	Easy	Moderate	Hard
SA-SSD [23]	93.23	84.30	81.36	-	-	-
CLOCs-PVCas [51]	92.78	85.94	**83.25**	93.48	91.98	89.48
PV-RCNN [54]	92.57	84.83	82.69	95.76	91.11	88.93
SA-SSD	93.14	84.65	81.86	96.56	92.84	90.36
SA-SSD+EBM	**95.45**	**86.83**	82.23	**96.60**	**92.92**	**90.43**
Rel. Improvement	+2.48%	+2.58%	+0.45%	+0.04%	+0.09%	+0.08%

Table 2. Results on KITTI val in terms of 3D and BEV AP. Our proposed detector consistently outperforms the SA-SSD baseline, and sets a new state-of-the-art for all but one of the metrics.

amples) and *val* (3 769 examples) splits. We train models exclusively on the *train* split and set hyperparameters using the *val* split. We report results both on *val*, and on the *test* split by submitting detections to the KITTI benchmark server. Following SA-SSD, we conduct experiments only on the car object class.

Evaluation Metrics On the KITTI benchmark server, models are evaluated in terms of average precision (AP) in both 3D and BEV. It considers three different difficulty levels (easy, moderate and hard), based on how far away and occluded objects are. AP is the area under the precision-recall curve, where a predicted bounding box is considered a true positive if its 3D/BEV IoU with a ground truth box exceeds a certain threshold. For cars, the threshold is set to 0.7 on the KITTI benchmark. Two predicted boxes with IoU of, e.g., 0.71 and 0.99 thus have identical effect on this metric. Since our main goal is to improve the accuracy of all predicted bounding boxes, we also report the AP for higher thresholds $\{0.75, 0.8, 0.85, 0.9\}$ on the *val* split. All reported AP values are computed using 40 recall positions.

4.2. Implementation Details

We utilize the open-source implementation and pre-trained model provided[1] by the SA-SSD authors. The feature map $h_3(x)$ that is produced by the backbone network is of shape $200 \times 176 \times 256$. We divide each y^{BEV} (7) into a regular 4×7 grid, meaning that the feature vector $h_4(x, y^{\text{BEV}}) \in \mathbb{R}^{7168}$. We process $c_z \in \mathbb{R}$ and $h \in \mathbb{R}$ with separate fully-connected layers (dimensions: $1 \rightarrow 16$, $16 \rightarrow 16$), generating $g_{c_z} \in \mathbb{R}^{16}$ and $g_h \in \mathbb{R}^{16}$. After concatenation, we thus obtain $h_5(x, y) \in \mathbb{R}^{7200}$. Finally, $h_5(x, y)$ is processed by three fully-connected layers of dimensions $7200 \rightarrow 1024$, $1024 \rightarrow 1024$, $1024 \rightarrow 1$. To train the DNN $f_\theta(x, y)$, i.e. the added fully-connected layers, we just replace the original detector loss with the NCE loss (Sec. 3.3). We also considered NCE+ with $\beta > 0$, but saw no clear improvements over NCE. We hypothesize this is because there is less inherent ambiguity in the annotation process of 3D bounding boxes than of 2D bounding boxes in images. As in [20], we set $K = 3$ with

[1] https://github.com/skyhehe123/SA-SSD

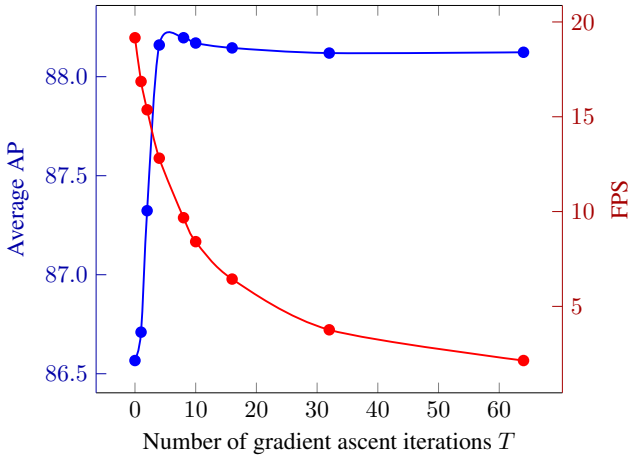

Figure 5. Impact of the number of gradient ascent iterations T in Algorithm 1 on detector performance (3D AP with 0.7 threshold, averaged over easy, moderate and hard) and detector inference speed (FPS), on KITTI *val*. Refinement with $T = 4$ iterations significantly improves the detector performance, while only decreasing the inference speed from 19.2 to 12.8 FPS.

$\sigma_1 = \sigma_3/4$, $\sigma_2 = \sigma_3/2$ for the noise distribution $q(y|y_i)$ (5). After ablation, optimizing 3D AP (moderate difficulty) on the *val* split, we set σ_3 differently for different components of the 3D box y (6). Specifically, $\sigma_3 = 0.25$ for (c_x, c_y), $\sigma_3 = 0.125$ for (c_z, h, w, l) and $\sigma_3 = 0.0625$ for ϕ. Following [19, 20], we also set $T = 10$ and $\eta = 0.5$ for gradient-based refinement (Algorithm 1). The step-length $\lambda = 0.0002$ was selected based on ablation.

4.3. 3DOD Results on KITTI

Results on KITTI *test* in terms of 3D and BEV AP (0.7 threshold) are found in Table 1. We mainly compare the performance of our EBM-based 3D object detector (SA-SSD+EBM) to the pre-trained SA-SSD it extends, and include other state-of-the-art detectors for reference. We also include the results for SA-SSD reported in the original paper [23], as these differ somewhat from those obtained with the provided pre-trained model. In Table 1, we observe that the added EBM and gradient-based refinement consistently improves the SA-SSD baseline across all metrics. We also observe that our SA-SSD+EBM detector achieves very competitive performance compared to previous methods.

Results on KITTI *val* in terms of 3D and BEV AP (0.7 threshold) are found in Table 2. There, we only include detectors for which AP values computed using 40 recall positions are available. In Table 2, we again observe that our EBM-based detector consistently outperforms the SA-SSD baseline. On KITTI *val*, our SA-SSD+EBM also sets a new state-of-the-art in terms of all but one of the metrics.

A further comparison of SA-SSD+EBM and the SA-SSD baseline is provided in Table 3. There, we report AP

	3D @ 0.75			3D @ 0.8			3D @ 0.85			3D @ 0.9		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
SA-SSD	84.48	73.91	70.99	60.89	50.08	47.37	24.29	19.58	18.05	2.06	1.58	1.33
SA-SSD+EBM	87.85	74.96	71.95	66.70	54.32	51.36	31.02	23.91	21.95	3.45	2.74	2.26
Rel. Improvement	+3.99%	+1.42%	+1.35%	+9.54%	+8.47%	+8.42%	+27.7%	+22.1%	+21.6%	+67.5%	+73.4%	+69.9%
	BEV @ 0.75			BEV @ 0.8			BEV @ 0.85			BEV @ 0.9		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
SA-SSD	95.41	87.47	84.79	87.12	79.07	74.65	61.53	54.15	50.39	17.48	15.71	14.58
SA-SSD+EBM	95.47	87.54	84.88	88.31	80.06	77.25	68.40	58.62	54.48	26.60	22.03	19.48
Rel. Improvement	+0.06%	+0.08%	+0.11%	+1.37%	+1.25%	+3.48%	+11.2%	+8.25%	+8.12%	+52.2%	+40.2%	+33.6%

Table 3. Results on KITTI val in terms of 3D and BEV AP for higher thresholds $\{0.75, 0.8, 0.85, 0.9\}$. Our SA-SSD+EBM detector consistently outperforms the SA-SSD baseline across all metrics, and the relative improvement increases with the AP threshold.

for higher thresholds $\{0.75, 0.8, 0.85, 0.9\}$ on KITTI *val*. We observe that the gradient-based refinement consistently improves detector performance across all metrics, and that the relative gain is larger for higher thresholds. Our approach thus also refines accurate bounding boxes even further, an effect not captured by the standard AP metrics.

4.4. Analysis of Inference Speed

The improved detection performance compared to SA-SSD comes with a decreased inference speed. On a single NVIDIA TITAN Xp GPU, SA-SSD runs at 19.2 FPS, while SA-SSD+EBM runs at 8.4 FPS for $T = 10$ gradient ascent iterations. We further analyze how the choice of T affects detector inference speed and performance in Figure 5. The performance is here given in terms of 3D AP (0.7 threshold) averaged over the three difficulty levels (easy, moderate, hard), on KITTI *val*. We observe that the choice $T = 4$ provides approximately equal performance compared to $T = 10$, while only decreasing the inference speed to 12.8 FPS. This trade-off between detector performance and inference speed could potentially be further improved by using fewer grid points in our RoIAlign variant, or by using a more lightweight energy prediction network branch. Our approach could also be very well-suited for offboard 3DOD [53], where inference speed is less of a concern. Exploring these directions is left for future work.

4.5. Analysis of Learned Distribution

For 3DOD from LiDAR point clouds, it can be inherently difficult to correctly predict the heading angle ϕ of a 3D bounding box y (6). This is because it is often difficult, when only given a point cloud, to distinguish between two otherwise identical cars which are heading in opposite directions. The true distribution $p(y|x)$ will thus often have two distinct modes, one at the true heading angle ϕ and one at $\phi+\pi$. In Figure 6, we visualize $f_\theta(x, y) \in \mathbb{R}$ as a function of $\Delta\phi$ when a predicted 3D bounding box y is rotated $\Delta\phi$ rad, demonstrating that our trained EBM $p(y|x;\theta)$ does indeed capture this inherent multi-modality in the true $p(y|x)$.

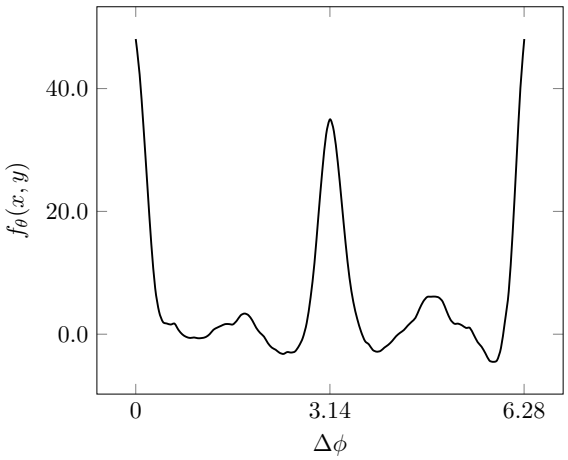

Figure 6. Visualization of the DNN scalar output $f_\theta(x, y)$ when a predicted 3D bounding box y (6) is rotated $\Delta\phi$ rad. The two distinct modes at $\Delta\phi = 0$ and $\Delta\phi = \pi$ demonstrate that the trained EBM $p(y|x;\theta)$ captures the inherent multi-modality in $p(y|x)$.

Future directions include investigating if the trained EBM $p(y|x;\theta)$ could be used to construct accurate estimates of prediction uncertainty, or provide other useful insights.

5. Conclusion

We applied conditional EBMs $p(y|x;\theta)$ to the task of 3D bounding box regression, thus extending the recent EBM-based regression approach from 2D to 3D object detection. By designing a differentiable pooling operator for 3D bounding boxes, we could integrate a conditional EBM $p(y|x;\theta)$ into the state-of-the-art 3D object detector SA-SSD. On the KITTI dataset, our approach consistently outperformed the SA-SSD baseline across all 3DOD metrics, and achieved highly competitive performance also compared to other state-of-the-art methods. By demonstrating the potential of EBM-based regression for highly accurate 3DOD, we hope that our work will encourage the research community to further explore the application of EBMs to 3DOD and other important regression tasks.

Acknowledgments This research was supported by the Swedish Foundation for Strategic Research via the project *ASSEMBLE*, the Knut and Alice Wallenberg Foundation via the *Wallenberg AI, Autonomous Systems and Software Program (WASP)*, and the *Kjell & Märta Beijer Foundation*.

References

[1] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 15509–15519, 2019. 3

[2] Fan Bao, Chongxuan Li, Kun Xu, Hang Su, Jun Zhu, and Bo Zhang. Bi-level score matching for learning energy-based latent variable models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[3] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003. 2

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020. 3

[5] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12093–12102, 2020. 1

[6] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7183–7192, 2020. 2, 4, 5

[7] Raul Diaz and Amit Marathe. Soft labels for ordinal regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[8] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[9] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 3266–3273, 2018. 2

[10] Di Feng, Lars Rosenbaum, Claudius Glaeser, Fabian Timm, and Klaus Dietmayer. Can we trust you? On calibration of a probabilistic object detector for autonomous driving. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Workshop*, 2019. 2

[11] Di Feng, Lars Rosenbaum, Fabian Timm, and Klaus Dietmayer. Leveraging heteroscedastic aleatoric uncertainties for robust real-time lidar 3d object detection. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 1280–1287, 2019. 2

[12] Ruiqi Gao, Yang Lu, Junpei Zhou, Song-Chun Zhu, and Ying Nian Wu. Learning generative convnets via multi-grid modeling and sampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9155–9164, 2018. 2

[13] Ruiqi Gao, Erik Nijkamp, Diederik P Kingma, Zhen Xu, Andrew M Dai, and Ying Nian Wu. Flow contrastive estimation of energy-based models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7518–7528, 2020. 2

[14] Jochen Gast and Stefan Roth. Lightweight probabilistic deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3369–3378, 2018. 1

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012. 1, 2, 5

[16] Ross B. Girshick. Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015. 4

[17] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9224–9232, 2018. 3

[18] Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[19] Fredrik K Gustafsson, Martin Danelljan, Goutam Bhat, and Thomas B Schön. Energy-based models for deep probabilistic regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 2, 3, 4, 5, 6

[20] Fredrik K Gustafsson, Martin Danelljan, Radu Timofte, and Thomas B Schön. How to train your energy-based model for regression. In *Proceedings of the British Machine Vision Conference (BMVC)*, September 2020. 2, 3, 4, 5, 6

[21] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 297–304, 2010. 3

[22] Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[23] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11873–11882, 2020. 1, 2, 3, 5, 6

[24] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017. 4

[25] Geoffrey Hinton, Simon Osindero, Max Welling, and Yee-Whye Teh. Unsupervised discovery of nonlinear structure using contrastive backpropagation. *Cognitive science*, 30(4):725–731, 2006. 2

[26] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua

Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations (ICLR)*, 2019. 3

[27] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. EPNet: Enhancing point features with image semantics for 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 5

[28] Peter J Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, pages 73–101, 1964. 1, 4

[29] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709, 2005. 3

[30] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–799, 2018. 2, 4

[31] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016. 3

[32] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8, 2018. 1

[33] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12697–12705, 2019. 1

[34] Stéphane Lathuilière, Pablo Mesejo, Xavier Alameda-Pineda, and Radu Horaud. A comprehensive analysis of deep regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2019. 1

[35] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006. 2

[36] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7345–7353, 2019. 1

[37] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 641–656, 2018. 1

[38] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017. 4

[39] Zhuang Ma and Michael Collins. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3698–3707, 2018. 3

[40] Osama Makansi, Eddy Ilg, Ozgun Cicek, and Thomas Brox. Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7144–7153, 2019. 1

[41] Gregory P Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12677–12686, 2019. 2

[42] Michael Meyer and Georg Kuschk. Automotive radar dataset for deep learning based 3d object detection. In *2019 16th European Radar Conference (EuRAD)*, pages 129–132. IEEE, 2019. 1

[43] Michael Meyer and Georg Kuschk. Deep learning based 3d object detection for automotive radar and camera. In *2019 16th European Radar Conference (EuRAD)*, pages 133–136. IEEE, 2019. 1

[44] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3111–3119, 2013. 3

[45] Andriy Mnih and Geoffrey Hinton. Learning nonlinear constraints with contrastive backpropagation. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, volume 2, pages 1302–1307. IEEE, 2005. 2

[46] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. In *International Conference on Machine Learning (ICML)*, pages 1751–1758, 2012. 3

[47] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run mcmc toward energy-based model. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5233–5243, 2019. 2

[48] Margarita Osadchy, Matthew L Miller, and Yann L Cun. Synergistic face detection and pose estimation with energy-based models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1017–1024, 2005. 2

[49] Hongyu Pan, Hu Han, Shiguang Shan, and Xilin Chen. Mean-variance loss for deep age estimation from a face. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5285–5294, 2018. 1

[50] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[51] Su Pang, Daniel Morris, and Hayder Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. 1, 5, 6

[52] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019. 5

[53] Charles R Qi, Yin Zhou, Mahyar Najibi, Pei Sun, Khoa Vo, Boyang Deng, and Dragomir Anguelov. Offboard 3D object detection from point cloud sequences. *arXiv preprint arXiv:2103.05073*, 2021. 7

[54] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10529–10538, 2020. 1, 4, 5, 6

[55] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. PointRCNN: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–779, 2019. 1, 4

[56] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 1, 4, 5

[57] Weijing Shi and Raj Rajkumar. Point-GNN: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1711–1719, 2020. 5

[58] Xuepeng Shi, Zhixiang Chen, and Tae-Kyun Kim. Distance-normalized unified representation for monocular 3d object detection. In *European Conference on Computer Vision (ECCV)*, pages 91–107. Springer, 2020. 1

[59] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1991–1999, 2019. 1

[60] Yee Whye Teh, Max Welling, Simon Osindero, and Geoffrey E Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4(Dec):1235–1260, 2003. 2

[61] Jianwen Xie, Yang Lu, Song-Chun Zhu, and Yingnian Wu. A theory of generative convnet. In *International Conference on Machine Learning (ICML)*, pages 2635–2644, 2016. 2

[62] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 3

[63] Bin Yang, Runsheng Guo, Ming Liang, Sergio Casas, and Raquel Urtasun. RadarNet: Exploiting radar for robust perception of dynamic objects. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 1

[64] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3DSSD: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11040–11048, 2020. 5

[65] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: Sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1951–1960, 2019. 1, 4, 5

[66] Jin Hyeok Yoo, Yeocheol Kim, Ji Song Kim, and Jun Won Choi. 3D-CVF: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 1, 5

[67] Dingfu Zhou, Jin Fang, Xibin Song, Liu Liu, Junbo Yin, Yuchao Dai, Hongdong Li, and Ruigang Yang. Joint 3d instance segmentation and object detection for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1839–1849, 2020. 5

[68] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4490–4499, 2018. 1