UPPSALA
UNIVERSITET

Accurate 3D Object Detection using Energy-Based Models

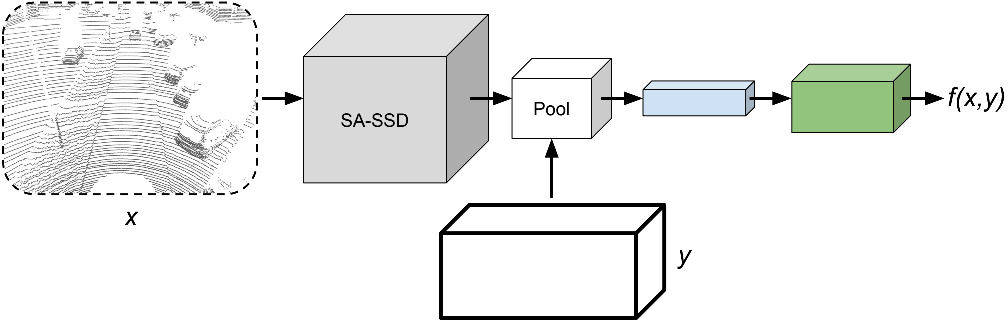
Fredrik K. Gustafsson
Uppsala University

Zenseact
January 29, 2021

Accurate 3D Object Detection using Energy-Based Models

Fredrik K. Gustafsson, Martin Danelljan (ETH Zürich), Thomas B. Schön (Uppsala University)

- arXiv: https://arxiv.org/abs/2012.04634
- Code (to be uploaded): https://github.com/fregu856/ebms_3dod
- Qualitative results: https://youtu.be/7JP6V818bh0
- These slides: http://www.fregu856.com



Energy-Based Models for Deep Probabilistic Regression
Fredrik K. Gustafsson, Martin Danelljan, Goutam Bhat, Thomas B. Schön
ECCV 2020

How to Train Your Energy-Based Model for Regression
Fredrik K. Gustafsson, Martin Danelljan, Radu Timofte, Thomas B. Schön
BMVC 2020

Accurate 3D Object Detection using Energy-Based Models
Fredrik K. Gustafsson, Martin Danelljan, Thomas B. Schön
Preprint

Outline

1. Energy-Based Models

2. Energy-Based Models for Regression

3. Energy-Based Models for 3D Object Detection

Outline

UPPSALA UNIVERSITET

1. Energy-Based Models

2. Energy-Based Models for Regression

3. Energy-Based Models for 3D Object Detection

Energy-based models [12] have a rich history within machine learning [19, 14, 9, 17].

An **energy-based model (EBM)** specifies a probability distribution $p(x;\theta)$ over $x \in \mathcal{X}$ directly via a parameterized scalar function $f_\theta : \mathcal{X} \to \mathbb{R}$:

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

Energy-based models [12] have a rich history within machine learning [19, 14, 9, 17].

An **energy-based model (EBM)** specifies a probability distribution $p(x;\theta)$ over $x \in \mathcal{X}$ directly via a parameterized scalar function $f_\theta : \mathcal{X} \to \mathbb{R}$:

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

By defining $f_\theta(x)$ using a deep neural network (DNN), $p(x;\theta)$ becomes expressive enough to learn practically any distribution from observed data.

Energy-based models [12] have a rich history within machine learning [19, 14, 9, 17].

An **energy-based model (EBM)** specifies a probability distribution $p(x;\theta)$ over $x \in \mathcal{X}$ directly via a parameterized scalar function $f_\theta : \mathcal{X} \to \mathbb{R}$:

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

By defining $f_\theta(x)$ using a deep neural network (DNN), $p(x;\theta)$ becomes expressive enough to learn practically any distribution from observed data.

EBMs have therefore become increasingly popular within computer vision in recent years, commonly being applied for various generative image modeling tasks [20, 3, 16, 2, 5, 15, 4, 18, 1].

An EBM specifies a probability distribution $p(x;\theta)$ directly via a parameterized scalar function $f_\theta(x)$,

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

where $f_\theta(x)$ commonly is defined using a DNN.

An EBM specifies a probability distribution $p(x;\theta)$ directly via a parameterized scalar function $f_\theta(x)$,

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

where $f_\theta(x)$ commonly is defined using a DNN.

The EBM $p(x;\theta) = e^{f_\theta(x)} / \int e^{f_\theta(\tilde{x})} d\tilde{x}$ is thus a highly expressive model that puts minimal restricting assumptions on the true distribution $p(x)$.

An EBM specifies a probability distribution $p(x;\theta)$ directly via a parameterized scalar function $f_\theta(x)$,

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

where $f_\theta(x)$ commonly is defined using a DNN.

The EBM $p(x;\theta) = e^{f_\theta(x)} / \int e^{f_\theta(\tilde{x})} d\tilde{x}$ is thus a highly expressive model that puts minimal restricting assumptions on the true distribution $p(x)$.

Drawback: the normalizing partition function $Z(\theta) = \int e^{f_\theta(\tilde{x})} d\tilde{x}$ is intractable, which complicates evaluating or sampling from $p(x;\theta)$.

An EBM specifies a probability distribution $p(x;\theta)$ directly via a parameterized scalar function $f_\theta(x)$,

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

where $f_\theta(x)$ commonly is defined using a DNN.

The EBM $p(x;\theta) = e^{f_\theta(x)} / \int e^{f_\theta(\tilde{x})} d\tilde{x}$ is thus a highly expressive model that puts minimal restricting assumptions on the true distribution $p(x)$.

Drawback: the normalizing partition function $Z(\theta) = \int e^{f_\theta(\tilde{x})} d\tilde{x}$ is intractable, which complicates evaluating or sampling from $p(x;\theta)$.

(Compare with normalizing flows which are easy to both evaluate and sample, but impose a specific structure on $p(x)$. EBMs instead prioritize maximum expressivity.)

The definition of an EBM $p(x;\theta)$,

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

includes the intractable $Z(\theta) = \int e^{f_\theta(\tilde{x})} d\tilde{x}$.

This complicates evaluating or sampling from $p(x;\theta)$.

The definition of an EBM $p(x;\theta)$,

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

includes the intractable $Z(\theta) = \int e^{f_\theta(\tilde{x})} d\tilde{x}$.

This complicates evaluating or sampling from $p(x;\theta)$.

In particular, EBMs are challenging to train. A variety of different approaches have therefore been explored in literature.

A very recent tutorial on the subject:

How to Train Your Energy-Based Models
Yang Song, Diederik P. Kingma
arXiv:2101.03288

Outline

UPPSALA UNIVERSITET

1. Energy-Based Models

2. Energy-Based Models for Regression

3. Energy-Based Models for 3D Object Detection

While EBMs recently had become increasingly popular within computer vision, they were basically only being employed for generative image modeling.

In **Energy-Based Models for Deep Probabilistic Regression**, we instead explored the application of EBMs to various regression problems (age estimation, head-pose estimation, 2D bounding box regression).

While EBMs recently had become increasingly popular within computer vision, they were basically only being employed for generative image modeling.

In **Energy-Based Models for Deep Probabilistic Regression**, we instead explored the application of EBMs to various regression problems (age estimation, head-pose estimation, 2D bounding box regression).

Regression: learn to predict a continuous target $y^\star \in \mathcal{Y} = \mathbb{R}^K$ from a corresponding input $x^\star \in \mathcal{X}$, given a training set $\mathcal{D}$ of i.i.d. input-target pairs, $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, $(x_i, y_i) \sim p(x, y)$.

We addressed this task by modelling the distribution $p(y|x)$ with a *conditional* EBM:

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

We addressed this task by modelling the distribution $p(y|x)$ with a *conditional* EBM:

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

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 partition function.

UPPSALA UNIVERSITET

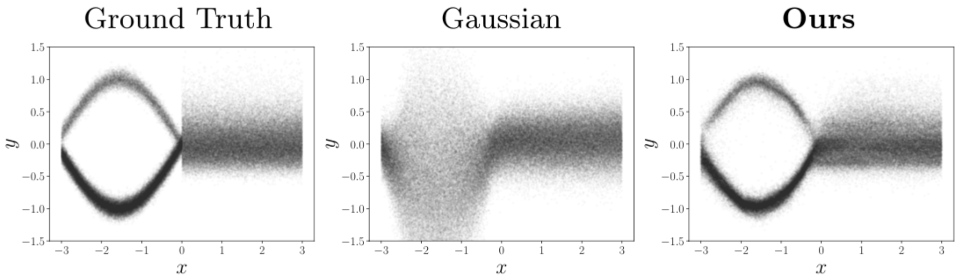
EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x; \theta)$:

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

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

The EBM $p(y|x;\theta)$ can learn complex target distributions directly from data:



EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

Given an input $x^\star$ at test time, we predict the target $y^\star$ by maximizing $p(y|x^\star;\theta)$:

$$y^\star = \underset{y}{\operatorname{argmax}}\, p(y|x^\star;\theta) = \underset{y}{\operatorname{argmax}}\, f_\theta(x^\star, y)$$

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

Given an input $x^\star$ at test time, we predict the target $y^\star$ by maximizing $p(y|x^\star;\theta)$:

$$y^\star = \underset{y}{\operatorname{argmax}}\, p(y|x^\star;\theta) = \underset{y}{\operatorname{argmax}}\, f_\theta(x^\star, y)$$

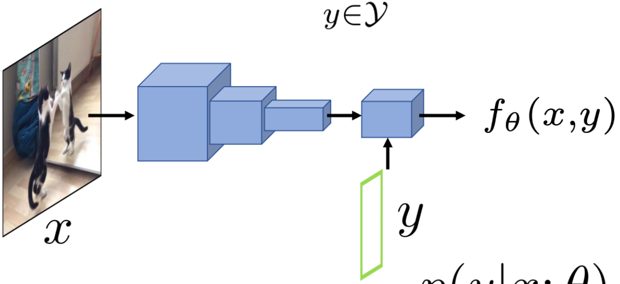
In practice, $y^\star = \operatorname{argmax}_y f_\theta(x^\star, y)$ is approximated by refining an initial estimate $\hat{y}$ via T steps of gradient ascent,

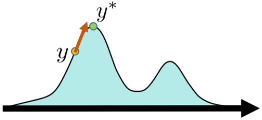
$$y \leftarrow y + \lambda \nabla_y f_\theta(x^\star, y),$$

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. Energy-Based Models for Regression - Prediction

$$y^* = \arg\max_{y \in \mathcal{Y}} p(y|x;\theta) = \arg\max_{y \in \mathcal{Y}} f_\theta(x, y)$$





$$p(y|x;\theta) = \frac{e^{f_\theta(x,y)}}{\int e^{f_\theta(x,\tilde{y})} \mathrm{d}\tilde{y}}$$

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

The DNN $f_\theta(x, y)$ can be trained using various methods for fitting a density $p(y|x;\theta)$ to observed data $\{(x_i, y_i)\}_{i=1}^N$.

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

The DNN $f_\theta(x, y)$ 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,

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

UPPSALA UNIVERSITET

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

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

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

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

The integral $\int e^{f_\theta(x_i,y)} dy$ is however intractable, preventing exact evaluation of $\mathcal{L}(\theta)$.

UPPSALA UNIVERSITET

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

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

The integral $\int e^{f_\theta(x_i,y)} dy$ is however intractable, preventing exact evaluation of $\mathcal{L}(\theta)$.

In **Energy-Based Models for Deep Probabilistic Regression**, we simply approximated this intractable integral using importance sampling.

UPPSALA UNIVERSITET

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

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

Importance sampling:

$$\begin{aligned}
-\log p(y_i|x_i;\theta) &= \log\left(\int e^{f_\theta(x_i,y)} dy\right) - f_\theta(x_i,y_i) \\
&= \log\left(\int \frac{e^{f_\theta(x_i,y)}}{q(y)} q(y) dy\right) - f_\theta(x_i,y_i) \\
&\approx \log\left(\frac{1}{M}\sum_{k=1}^{M} \frac{e^{f_\theta(x_i,y^{(k)})}}{q(y^{(k)})}\right) - f_\theta(x_i,y_i), \quad y^{(k)} \sim q(y).
\end{aligned}$$

UPPSALA UNIVERSITET

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

Various alternative techniques could however also be employed to train the DNN $f_\theta(x, y)$, including noise contrastive estimation (NCE) [6, 13] and score matching [10].

UPPSALA UNIVERSITET

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x;\theta)$:

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

Various alternative techniques could however also be employed to train the DNN $f_\theta(x, y)$, including noise contrastive estimation (NCE) [6, 13] and score matching [10].

In **How to Train Your Energy-Based Model for Regression**, we therefore studied in detail how EBMs should be trained specifically for regression problems.

EBMs for Regression: train a DNN $f_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to predict a scalar value $f_\theta(x, y)$, then model $p(y|x)$ with the conditional EBM $p(y|x; \theta)$:

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

Various alternative techniques could however also be employed to train the DNN $f_\theta(x, y)$, including noise contrastive estimation (NCE) [6, 13] and score matching [10].

In **How to Train Your Energy-Based Model for Regression**, we therefore studied in detail how EBMs should be trained specifically for regression problems.

We compared six methods on the task of 2D bounding box regression, and concluded that a simple extension of NCE should be considered the go-to training method.

UPPSALA UNIVERSITET

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

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

Noise contrastive estimation (NCE) entails learning to discriminate between observed data examples and samples drawn from a noise distribution.

UPPSALA UNIVERSITET

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

Noise contrastive estimation (NCE) entails learning to discriminate between observed data examples and samples drawn from a noise distribution.

Specifically, the DNN $f_\theta(x, y)$ is trained by minimizing the loss $J(\theta) = -\frac{1}{N}\sum_{i=1}^{N} J_i(\theta)$,

$$J_i(\theta) = \log \frac{\exp\{f_\theta(x_i, y_i^{(0)}) - \log q(y_i^{(0)}|y_i)\}}{\sum_{m=0}^{M} \exp\{f_\theta(x_i, y_i^{(m)}) - \log q(y_i^{(m)}|y_i)\}},$$

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.

2. Energy-Based Models for Regression - Training using NCE

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

$$y_i^{(0)} \triangleq y_i, \quad \{y_i^{(m)}\}_{m=1}^{M} \sim q(y|y_i) \text{ (noise distribution)}.$$

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

$$y_i^{(0)} \triangleq y_i, \quad \{y_i^{(m)}\}_{m=1}^{M} \sim q(y|y_i) \text{ (noise distribution).}$$

Effectively, $J(\theta)$ is the softmax cross-entropy loss for a classification problem with $M+1$ classes (which of the $M+1$ values $\{y_i^{(m)}\}_{m=0}^{M}$ is the true target y_i?).

UPPSALA UNIVERSITET

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

$$y_i^{(0)} \triangleq y_i, \quad \{y_i^{(m)}\}_{m=1}^{M} \sim q(y|y_i) \text{ (noise distribution)}.$$

Effectively, $J(\theta)$ is the softmax cross-entropy loss for a classification problem with $M+1$ classes (which of the $M+1$ values $\{y_i^{(m)}\}_{m=0}^{M}$ is the true target y_i?).

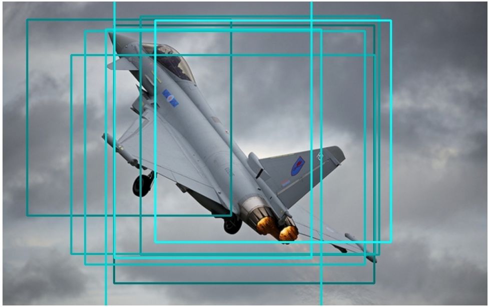
A simple yet effective choice for the noise distribution $q(y|y_i)$ is a mixture of K Gaussians centered at y_i,

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

UPPSALA UNIVERSITET

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

$$y_i^{(0)} \triangleq y_i, \quad \{y_i^{(m)}\}_{m=1}^{M} \sim q(y|y_i) \text{ (noise distribution)}.$$



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

$$y_i^{(0)} \triangleq y_i, \quad \{y_i^{(m)}\}_{m=1}^{M} \sim q(y|y_i) \text{ (noise distribution)}.$$

Outline

1. Energy-Based Models
2. Energy-Based Models for Regression
3. Energy-Based Models for 3D Object Detection

In **Accurate 3D Object Detection using Energy-Based Models**, we extend our energy-based regression approach from 2D to 3D bounding box regression.

In **Accurate 3D Object Detection using Energy-Based Models**, we extend our energy-based regression approach from 2D to 3D bounding box regression.

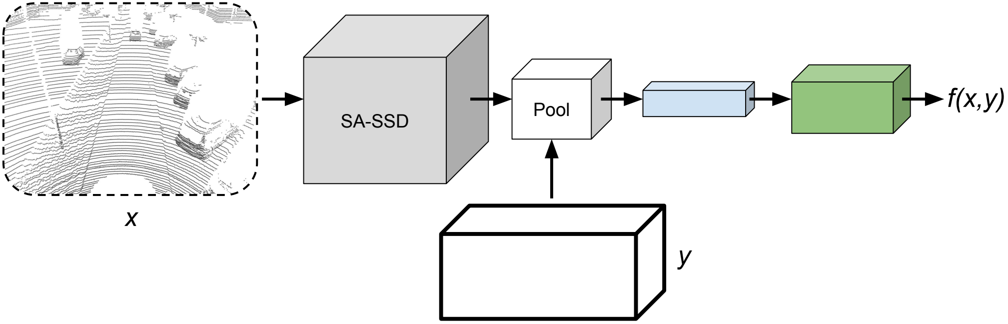
This is achieved by designing a differentiable pooling operator for 3D bounding boxes y, and adding an extra network branch to the state-of-the-art 3D object detector SA-SSD [7].

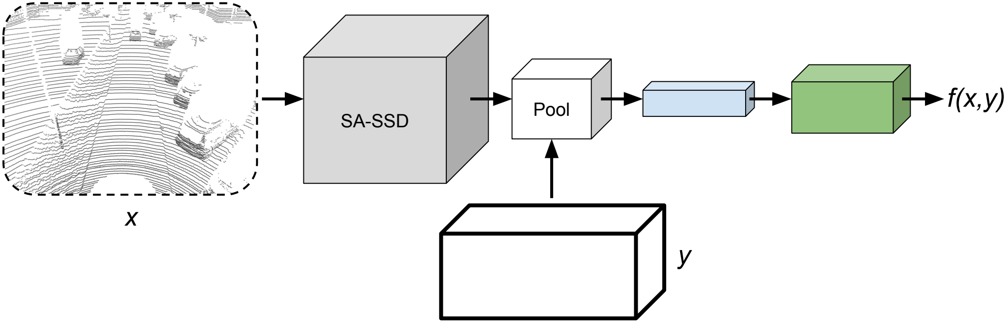
In **Accurate 3D Object Detection using Energy-Based Models**, we extend our energy-based regression approach from 2D to 3D bounding box regression.

This is achieved by designing a differentiable pooling operator for 3D bounding boxes y, and adding an extra network branch to the state-of-the-art 3D object detector SA-SSD [7].

We evaluate our proposed detector on the KITTI dataset and consistently outperform the SA-SSD baseline detector across all 3D object detection (3DOD) metrics.

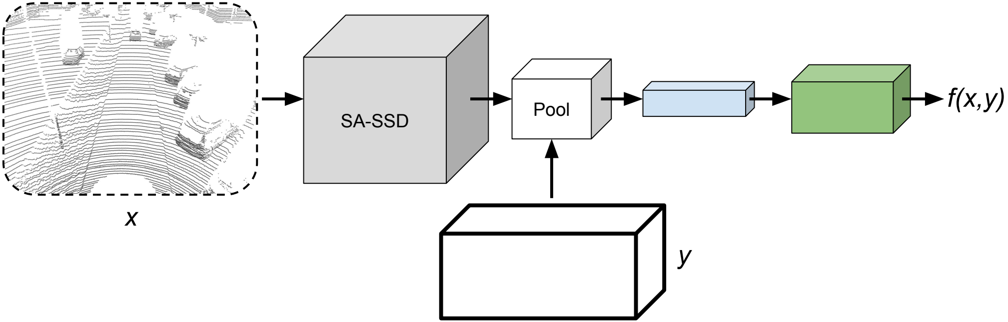
3. Energy-Based Models for 3D Object Detection





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 SA-SSD 3D object detector.

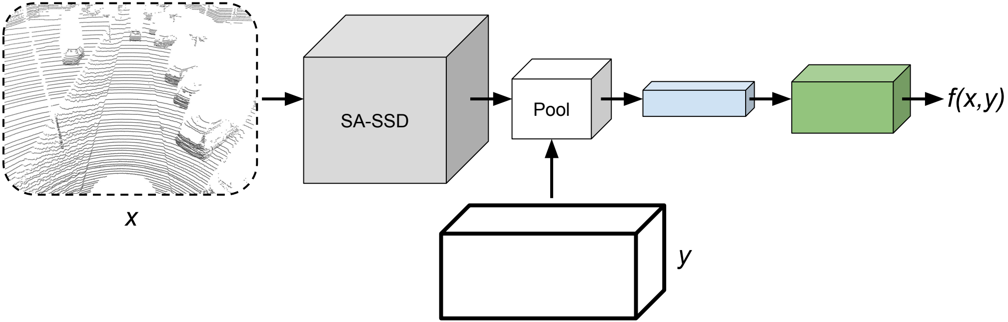
3. Energy-Based Models for 3D Object Detection

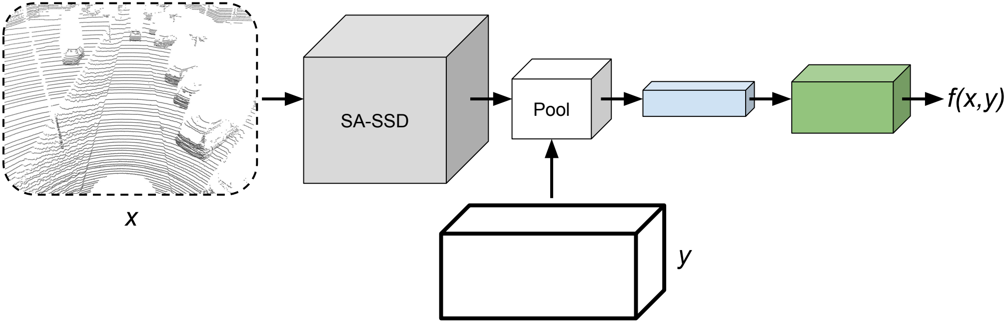


We integrate a conditional EBM $p(y|x;\theta) = e^{f_\theta(x,y)} / \int e^{f_\theta(x,\tilde{y})} d\tilde{y}$ into the SA-SSD 3D object detector.

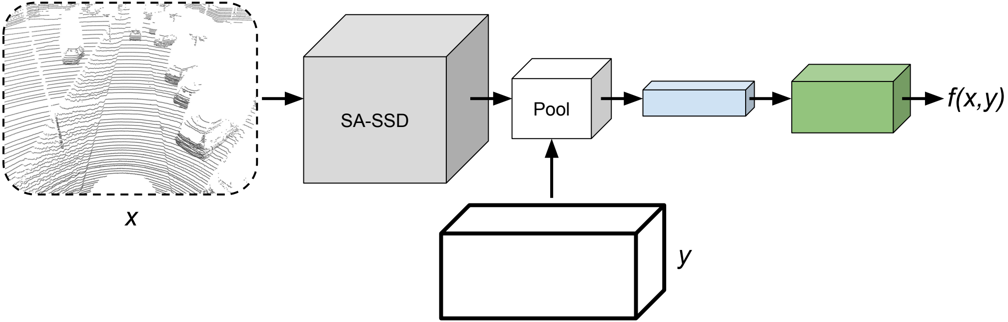
We design a differentiable pooling operator that, given a 3D bounding box y, extracts a feature vector from the SA-SSD output. This feature vector is processed by fully-connected layers, outputting $f_\theta(x, y) \in \mathbb{R}$.

3. Energy-Based Models for 3D Object Detection





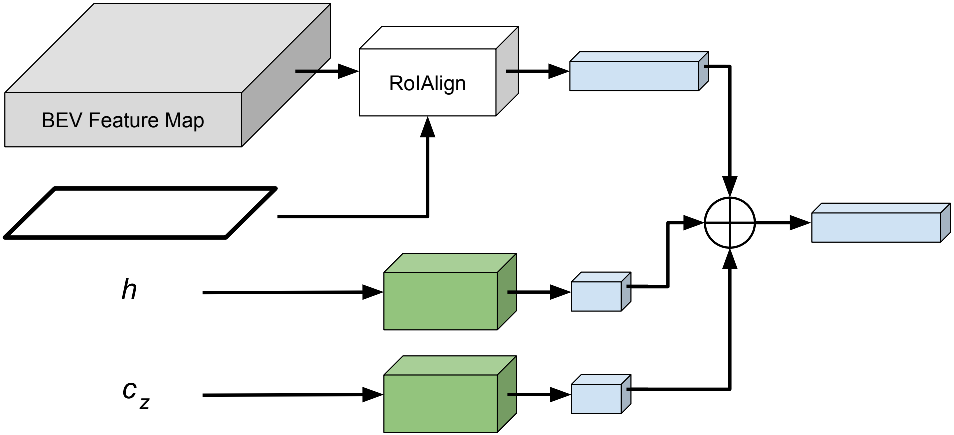
The differentiable pooling operator is required when using gradient ascent to maximize the EBM $p(y|x;\theta)$ at test-time, as this requires the scalar DNN output $f_\theta(x,y)$ to be differentiable w.r.t. the 3D bounding box y.



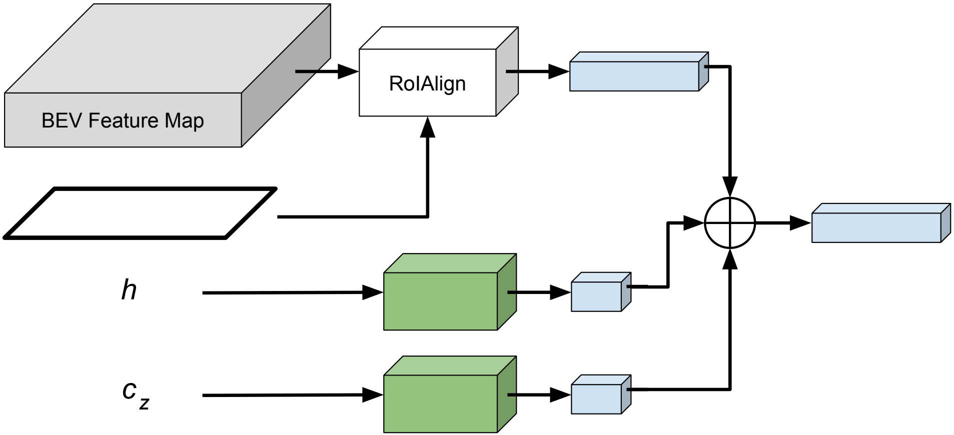
The differentiable pooling operator is required when using gradient ascent to maximize the EBM $p(y|x;\theta)$ at test-time, as this requires the scalar DNN output $f_\theta(x, y)$ to be differentiable w.r.t. the 3D bounding box y.

For 2D bounding boxes, this was achieved by applying a pooling operator [11] on image features, but this technique is not directly applicable to 3D bounding boxes.

UPPSALA UNIVERSITET

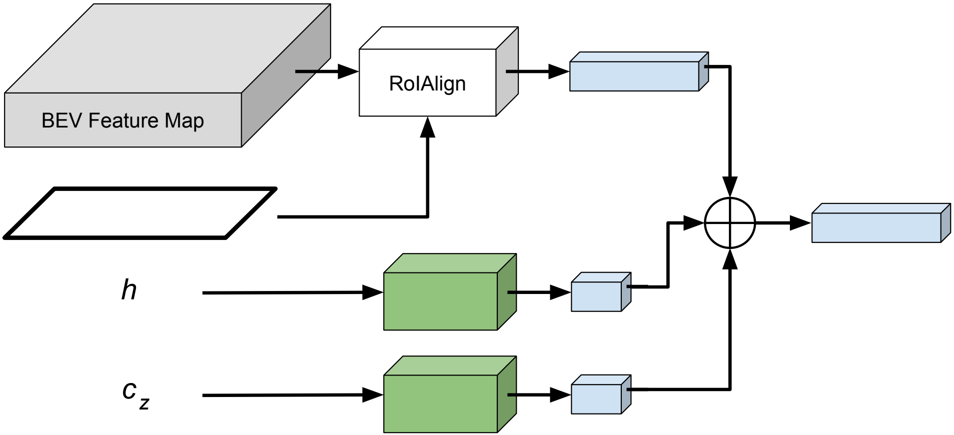


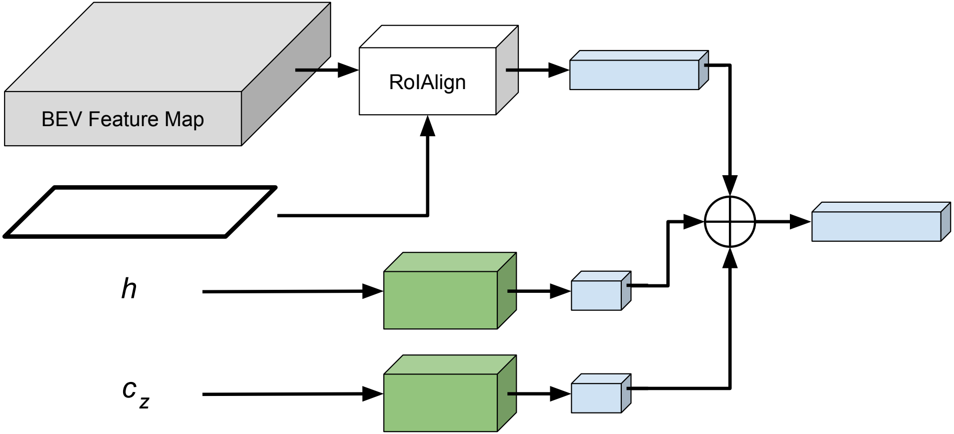
3. Energy-Based Models for 3D Object Detection - Pooling



The BEV version y^{BEV} of the 3D bounding box y is pooled with the BEV feature map produced by SA-SSD, extracting a feature vector. Since y^{BEV} is an oriented 2D box and not necessarily axis-aligned, we here employ a modified variant of RoIAlign [8].

3. Energy-Based Models for 3D Object Detection - Pooling





The z coordinate c_z and height h of the 3D bounding box y are processed by two small fully-connected layers, extracting a feature vector each. Finally, all three feature vectors are concatenated.

The extra fully-connected layers 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.

The extra fully-connected layers 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 previously described. We employ the same training parameters (batch size, data augmentation etc.) as for SA-SSD, only replacing the original detector loss with the NCE loss.

At test-time, the input LiDAR point cloud $x^\star$ is first processed by the SA-SSD detector. SA-SSD outputs a set $\{(\hat{y}_i, s_i)\}_{i=1}^{D}$ of D detections, where $\hat{y}_i$ is a 3D bounding box and s_i is the associated classification confidence score.

At test-time, the input LiDAR point cloud $x^\star$ is first processed by the SA-SSD detector. SA-SSD outputs a set $\{(\hat{y}_i, s_i)\}_{i=1}^{D}$ of D detections, where $\hat{y}_i$ is a 3D bounding box 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, 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 $f_\theta(x^\star, y)$.

At test-time, the input LiDAR point cloud $x^\star$ is first processed by the SA-SSD detector. SA-SSD outputs a set $\{(\hat{y}_i, s_i)\}_{i=1}^{D}$ of D detections, where $\hat{y}_i$ is a 3D bounding box 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, 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 $\{(y_i, s_i)\}_{i=1}^{D}$.

3. Energy-Based Models for 3DOD - Detector Inference

UPPSALA UNIVERSITET

Algorithm 1 Gradient-based refinement.

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

for $i = 1, \dots, D$ **do**
  $y \leftarrow \hat{y}_i$.
  for $t = 1, \dots, T$ **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 $\{y_i\}_{i=1}^{D}$.

TABLE I

RESULTS ON KITTI TEST IN TERMS OF 3D AND BEV AP.

	3D @ 0.7			BEV @ 0.7		
	Easy	Moderate	Hard	Easy	Moderate	Hard
Part-A^2 [2]	87.81	78.49	73.51	91.70	87.79	84.61
SERCNN [64]	87.74	78.96	74.30	94.11	88.10	83.43
EPNet [65]	89.81	79.28	74.59	94.22	88.47	83.69
Point-GNN [66]	88.33	79.47	72.29	93.11	89.17	83.90
3DSSD [67]	88.36	79.57	74.55	92.66	89.02	85.86
STD [1]	87.95	79.71	75.09	94.74	89.19	86.42
SA-SSD [24]	88.75	79.79	74.16	95.03	**91.03**	85.96
3D-CVF [14]	89.20	80.05	73.11	93.52	89.56	82.45
CLOCs-PVCas [13]	88.94	80.67	**77.15**	93.05	89.80	**86.57**
PV-RCNN [3]	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%

3. Energy-Based Models for 3DOD - Results

UPPSALA UNIVERSITET

TABLE II

RESULTS ON KITTI VAL IN TERMS OF 3D AND BEV AP.

	3D @ 0.7			BEV @ 0.7		
	Easy	Moderate	Hard	Easy	Moderate	Hard
SA-SSD [24]	93.23	84.30	81.36	-	-	-
CLOCs-PVCas [13]	92.78	85.94	**83.25**	93.48	91.98	89.48
PV-RCNN [3]	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%

UPPSALA UNIVERSITET

TABLE III

FURTHER COMPARISON OF OUR PROPOSED DETECTOR AND THE SA-SSD BASELINE ON KITTI VAL.

	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%

3. Energy-Based Models for 3DOD - Analysis of Inference Speed

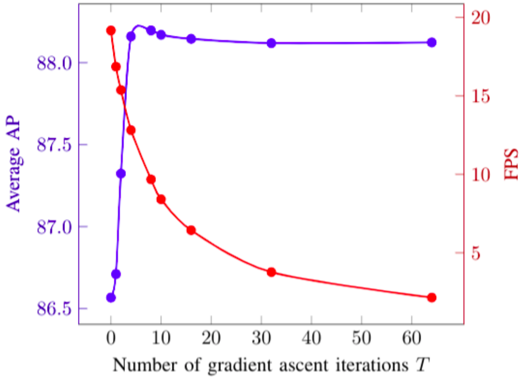


Fig. 5. Impact of the number of gradient ascent iterations T on detector performance (3D AP with 0.7 threshold, averaged over easy, moderate and hard) and detector inference speed (FPS), on KITTI *val*.

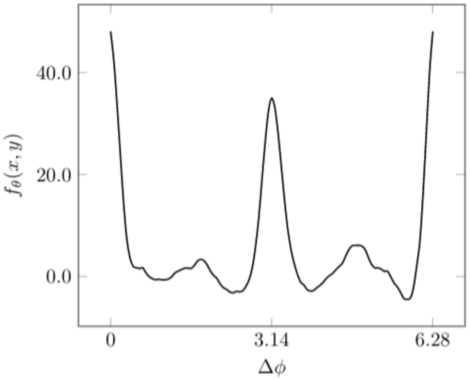


Fig. 6. Visualization of the DNN scalar output $f_\theta(x,y)$ when a predicted 3D bounding box y (6) is rotated $\Delta\phi$ rad, demonstrating that the trained EBM $p(y|x;\theta)$ captures the inherent multi-modality in $p(y|x)$.

[1] F. Bao, C. Li, K. Xu, H. Su, J. Zhu, and B. Zhang. Bi-level score matching for learning energy-based latent variable models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

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

[3] R. Gao, Y. Lu, J. Zhou, S.-C. Zhu, and Y. 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.

[4] R. Gao, E. Nijkamp, D. P. Kingma, Z. Xu, A. M. Dai, and Y. N. 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.

[5] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, D. Duvenaud, M. Norouzi, and K. Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations (ICLR)*, 2020.

[6] M. Gutmann and A. 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.

[7] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. 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.

[8] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[9] G. Hinton, S. Osindero, M. Welling, and Y.-W. Teh. Unsupervised discovery of nonlinear structure using contrastive backpropagation. *Cognitive science*, 30(4):725–731, 2006.

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

[11] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–799, 2018.

[12] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[13] Z. Ma and M. 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.

[14] A. Mnih and G. 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.

[15] E. Nijkamp, M. Hill, T. Han, S.-C. Zhu, and Y. N. Wu. On the anatomy of MCMC-based maximum likelihood learning of energy-based models. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

[16] E. Nijkamp, M. Hill, S.-C. Zhu, and Y. N. 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.

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

[18] B. Pang, T. Han, E. Nijkamp, S.-C. Zhu, and Y. N. Wu. Learning latent space energy-based prior model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[19] Y. W. Teh, M. Welling, S. Osindero, and G. E. Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4(Dec):1235–1260, 2003.

[20] J. Xie, Y. Lu, S.-C. Zhu, and Y. Wu. A theory of generative convnet. In *International Conference on Machine Learning (ICML)*, pages 2635–2644, 2016.

Contact

Fredrik K. Gustafsson, Uppsala University

fredrik.gustafsson@it.uu.se

www.fregu856.com