



HAL
open science

Learning Point Processes and Convolutional Neural Networks for object detection in satellite images

Jules Mabon, Mathias Ortner, Josiane Zerubia

► **To cite this version:**

Jules Mabon, Mathias Ortner, Josiane Zerubia. Learning Point Processes and Convolutional Neural Networks for object detection in satellite images. 2023. hal-04250540v1

HAL Id: hal-04250540

<https://inria.hal.science/hal-04250540v1>

Preprint submitted on 19 Oct 2023 (v1), last revised 15 Oct 2024 (v3)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Learning Point Processes and Convolutional Neural Networks for object detection in satellite images^{*}

Jules Mabon^{a,*}, Mathias Ortner^b, Josiane Zerubia^a

^a*Inria, Université Côte d'Azur, Sophia-Antipolis, France*

^b*Airbus Defense and Space, Toulouse, France*

Abstract

Convolutional Neural Networks have shown great results for object detection tasks by learning texture and pattern extraction filters. However, object-level interactions are harder to grasp without increasing the complexity of the architectures. On the other hand Point Process models propose to solve the detection on the configuration of objects as a whole, allowing to factor-in the image data, and the objects' interaction priors. In this paper we propose combining the information extracted by a CNN with priors on objects within a Markov Marked Point Process framework. We also demonstrate a method to learn the parameters of this Energy Based Model. We apply this model to the detection of small vehicles in optical satellite imagery, where the image information needs to be complemented with object interaction priors because of noise and small object sizes. The code will be made available at github.com/Ayana-Inria.

Keywords: Point Process, Convolutional Neural Networks, Energy Based Models, Remote Sensing

^{*}This document is the results of an Inria/Airbus Defense and Space research project funded by BPI France within the LiChIE project. The authors are also thankful to the OPAL infrastructure from Université Côte d'Azur for providing computational resources and support.

^{*}Corresponding author

Email address: jules.mabon@inria.fr (Jules Mabon)

1. Introduction

While object detection has been thoroughly studied for the last 20 years [1], detection of small objects in optical satellite images remains a challenging task: With the limited spatial resolution (between 0.25 and 0.5 m/pixel) objects such as cars can be only a few pixels wide. Moreover, this tiny object scattering density varies greatly, and when closely packed, instances might be difficult to differentiate. However, as the geometrical configuration of one object is often dependent on its neighbors, we can leverage these priors to improve the detection results.

Methods based on Convolutional Neural Networks (CNN) such as Faster R-CNN [2], YOLO [3] or RetinaNet [4] propose to detect objects in “natural” images. Some of these approaches first extract features through convolutions, then propose a series of boxes (anchors) that are refined by regression afterwards [5, 6]. The specificities of remote sensing data (high number of small objects), motivates the use of anchor-free method, relying on heatmap (probability of object center) inference for oriented vehicle detection [7, 8], or ship detection [9] (here approximating the heatmap through a vector field).

The majority of these approaches do not consider the interactions between objects more than the non-superposition introduced by the post-processing non-maximum suppression step. While CNNs are efficient at extracting texture and local level information [10], the long range relationships are only caught when increasing the amount of pooling layers or by introducing dense connections in the network¹.

Moreover, lower resolution reduces the information from which to infer object positions; [11] proposes to use super-resolution to detect ships, while [12] makes use of the temporal information.

Previous works on Point Process [13] demonstrate how to perform detection

¹Attention mechanisms can compute long range interactions, but greatly increase the complexity and number of parameters of the CNN model

28 while relying on stochastic geometry model to jointly solve the detection and
29 selection of objects with priors. Point process approaches model the probability
30 on the whole set of points in the image, thus taking into account the interactions
31 between objects. These approaches rely on decomposing the point process den-
32 sity into a data term which measures the correspondence of the points against
33 the image, and some prior terms, that measure the coherence of the point con-
34 figuration itself. These approaches have shown good results for detection tasks
35 in images with many small objects, such as biological imagery [14, 15] or remote
36 sensing for road [15, 16] building extraction [17] or tree detection [15].

37 Previous methods make use of data terms built upon image contrast mea-
38 sures between the interior and exterior of the object [14, 16, 18] or image gradi-
39 ent [19], which perform great in images with clear contrast between objects and
40 their background. However, in the case of optical satellite images, backgrounds
41 diversity, variable visual aspect of objects, and inconsistent illumination make
42 these contrast measures unreliable.

43 We propose leveraging the pattern matching capabilities of CNNs to build
44 a data term for a point process model, while performing regularization on the
45 configuration of points, with the priors on the point process. Moreover, we
46 estimate parameters of the point process model using a contrastive divergence
47 approach based on Energy Base Model (EBM) learning [20, 21].

48 **2. Point Process for object detection**

49 *2.1. Markov Marked Point Process*

50 We consider the image space as $S \subset \mathbb{R}^2$. A configuration of points Y is
51 a finite non-ordered set of elements of S : $Y = \{y^1, y^2, \dots\}$. We denote the
52 space of all possible configurations as $\mathcal{Y} = \cup_{n=0}^{\infty} S^n$. A Point Process is then the
53 application from a probability space to \mathcal{Y} , such as for any Borel subset $A \subseteq S$,
54 the number of points in A , $N(A)$, is a finite random variable.

55 A Marked Point Process considers configurations in the space $\mathcal{Y} = \cup_{n=0}^{\infty} (S \times$
56 $M)^n$, where M is the space of marks, which encodes information about each

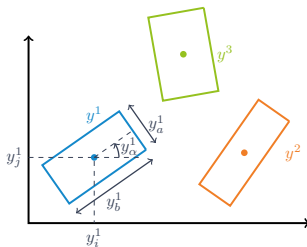


Figure 1: Example configuration with three points; $Y = \{y^1, y^2, y^3\}$.

57 point (e.g., geometric, or semantic information). In our case, an object $y \in Y$
 58 is composed of coordinates i, j in S , and three marks that describe a rectan-
 59 gle: Width a , length b and angle α (see Fig. 1), thus $M = [a_{\min}, a_{\max}] \times$
 60 $[b_{\min}, b_{\max}] \times [\alpha_{\min}, \alpha_{\max}]$. We denote respectively the components of point y as
 61 $(y_i, y_j, y_a, y_b, y_\alpha)$.

We model configurations of points Y as the realization of a non-uniform
 Marked Point Process. The density of which is defined by h relative to the
 uniform Marked Point Process [13]. The model of selection and interaction of
 points derives from an energy U , through a non-normalized Gibbs density:

$$h(Y) \propto \exp(-U(Y)), \quad (1)$$

$$U(Y) = \sum_{y \in Y} V(y|\mathcal{N}_y). \quad (2)$$

62 With $V(y|\mathcal{N}_y)$ the energy of point y given the neighborhood of y in Y denoted
 63 \mathcal{N}_y , the formulation in Eq. 2 ensures the Point Process is Markovian; A Point
 64 Process is Markovian under the relationship² \sim , if and only if $\forall Y \in \mathcal{Y}$ such that
 65 $h(Y) > 0$:

- 66 1. $\forall Y' \subset Y, h(Y') > 0$
- 67 2. $\forall u \in S \times M, h(Y \cup \{u\})/h(Y)$ only depends on u and its neighbors ac-
 68 cording to \sim (i.e., $\{y \in Y, y \sim u\}$)

²Note that relationship \sim and the neighborhood relationship that defines \mathcal{N}_y are not the same. However, the most restrictive relation \sim that ensures Markovianity depends on \mathcal{N}_y .

69 In the rest of the paper, we consider a Markov Marked Point Process, referred
 70 to as Point Process (PP) for simplicity.

71 *2.2. Point process for object detection*

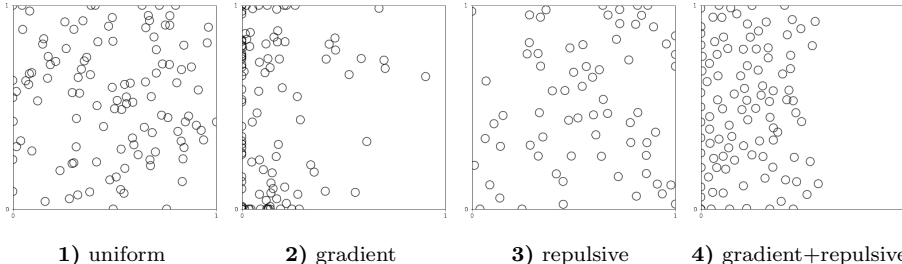


Figure 2: Point processes (without marks, $y = (y_i, y_j)$) derived from simple energies, for each example $k = 1, \dots, 4$, the configuration energy is $U(Y) = \sum_{y \in Y} V_k(y | \dots)$, with $d(y, y')$ the Euclidean distance between y and y' : **1)** $V_1(y) \propto 1$, **2)** $V_2(y) \propto y_i$, **3)** $V_3(y | \mathcal{N}_y) \propto \min_{y' \in \mathcal{N}_y} d(y, y')$, **4)** $V_4(y | \mathcal{N}_y) \propto V_2(y) + V_3(y | \mathcal{N}_y)$.

72 When performing object detection, the image provides information about
 73 the location and properties of objects, while prior knowledge on the objects of
 74 interest complements this information. Thus, the density and energy in Eq.
 75 1 take into account the image X to become respectively $h(Y|X)$, $U(Y|X)$.
 76 The *best* configuration for an image (or observation) — in the sense of the
 77 maximum *a posteriori* (i.e., maximizing $h(Y|X)$) — is the configuration Y^*
 78 with the lowest energy $U(Y|X)$ for a given image X . We denote the estimate
 79 of this configuration \hat{Y} .

80 The energy formulation of the point process allows compositing several sim-
 81 ple point behaviors/distributions into one model. We illustrate the composition
 82 of several simple point distributions through the addition of energy terms in
 83 Figure 2.

More generally: For every point $y \in Y$ we compute a set of N_V energy terms $V_k(y|X, \mathcal{N}_y)$, $k = 1, \dots, N_V$ and combine those — per point — as a weighted

sum[16, 17, 18]:

$$\begin{aligned}
 U(Y|X, \theta) &= \sum_{y \in Y} \theta_{w,0} + \sum_{k=1}^{N_V} \theta_{w,k} V_k(y|X, \mathcal{N}_y, \theta) \\
 &= \sum_{y \in Y} V(y|X, \mathcal{N}_y, \theta)
 \end{aligned} \tag{3}$$

84 with $V(y|X, \mathcal{N}_y, \theta)$ the energy contribution of a single object y , weighted sum
 85 of its sub-energy terms V_k .

86 The neighborhood of one point y in Y , for Euclidean distance $d(\cdot, \cdot)$, is
 87 defined as $\mathcal{N}_y^Y = \{\tilde{y} \in Y | \tilde{y} \neq y, d(y, \tilde{y}) < d_{\max}\}$, here often abbreviated \mathcal{N}_y .
 88 The set θ , contains the parameters of the energy model (e.g., weights θ_w) that
 89 need to be set before inferring \hat{Y} . The weights in θ_w (and other parameters in θ)
 90 can be either set manually or inferred by linear programming [18, 22]. Notation
 91 $\hat{\theta}$ represents the estimated parameters, the inference of which will be discussed
 92 in Section 4.

93 Finally, the classical denomination of energy terms goes as follows:

- 94 • *Prior terms* (of the form $V_k(y|\mathcal{N}_y, \theta)$) that only depend on y and its neigh-
 95 borhood \mathcal{N}_y ; they measure the coherence of the configuration itself con-
 96 sidering the known properties of the objects of interest (e.g., objects don't
 97 overlap).
- 98 • *Data terms* ($V_k(y|X, \theta)$) are function of y and the image X (also referred
 99 as observation in remote sensing); these terms measure the correspondence
 100 of the points with the image.

101 3. Energy model

102 The energy model returns a scalar energy value given an image X , parame-
 103 ters θ , and a configuration Y in \mathcal{Y} . This value is computed as a sum of energy
 104 terms for each point (Eq. 3). In this section, we define the multiple energy
 105 terms V_k : First the *data terms*, then the *prior terms*. We illustrate the pipeline
 106 resulting from the energy model in Figure 3.

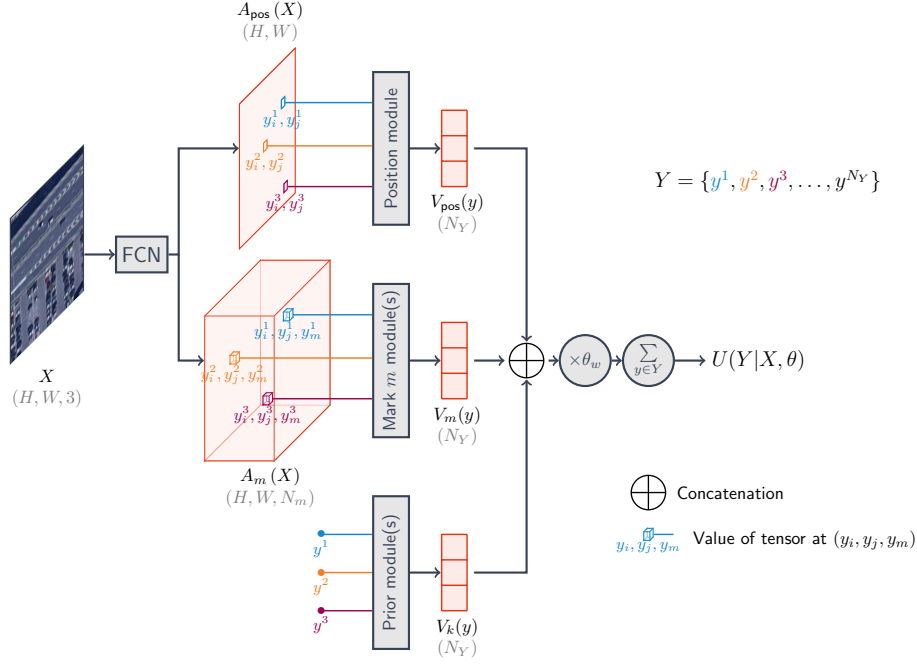


Figure 3: Simplified energy model, with only one mark and prior displayed. With FCN: Fully Convolutional Network, e.g., Unet [23].

107 *3.1. Data terms from CNNs*

108 Classical likelihood measurements [14, 16, 18] are based on contrast mea-
 109 sures, that are crafted to fit each application. Moreover, these measures rely
 110 heavily on the high contrast between objects and their background along with
 111 limited background diversity. We illustrate the limitation of those in Section
 112 6.2.

113 On the other hand CNNs have shown to be very efficient at extracting fea-
 114 tures from images for object detection and classification. In the following sec-
 115 tion, we will show how to interpret a CNN based object detector output to
 116 get an energy that measures the fitness of a configuration against an image. It
 117 allows us to go past the contrast measure design by utilizing a pre-trained CNN
 118 output.

119 *3.1.1. Position likelihood term*

120 CNN based object detection methods such as [7, 8, 24] make use of a heatmap
 121 to find object centers. This object center probability map is obtained by passing
 122 the image X of shape $H, W, 3$ (here with 3 color channels), into a fully convo-
 123 lutional model such as Unet [23]. It outputs a tensor $A_{\text{pos}}(X) \in \mathbb{R}^{H \times W}$, from
 124 which a probability-like measure of an object center at coordinates (y_i, y_j) is
 125 obtained as $p(y_i, y_j|X) = \sigma(A_{\text{pos}}(X)[y_i, y_j])$ where $\sigma(\cdot)$ is the sigmoid function,
 126 and $A_{\text{pos}}(X)[y_i, y_j]$ the value of map $A_{\text{pos}}(X)$ at coordinates (y_i, y_j) .

We define the position energy as such (see Appendix A.1 for details):

$$V_{\text{pos}}(y_i, y_j|X, \theta) = \ln(1 + \exp(-A_{\text{pos}}(X)[y_i, y_j] + \theta_{t,\text{pos}})) \quad (4)$$

127 With $\theta_{t,\text{pos}}$ a scalar threshold, allowing to move the inflection point of the
 128 softplus function $x \mapsto \ln(1 + \exp(-x))$.

129 As the raster map $A_{\text{pos}}(X)$ defines energies over discrete pixel positions, and
 130 coordinates $(y_i, y_j) \in S \subset \mathbb{R}^2$ are continuous, we perform bilinear interpolation
 131 to get energy $V_{\text{pos}}(y_i, y_j|X)$ for every location in S .

132 *3.1.2. Mark likelihood term*

For every mark m (here $m \in \{a, b, \alpha\}$), we define the mark data term
 $V_m(y_m|y_i, y_j, X)$ from the output of a CNN. Given a classification model that
 outputs, for a given position (y_i, y_j) and image X , the probability distribution
 over the N_m discretized mark values $m_k, k = 1, \dots, N_m$, for every discrete value
 m_k we have:

$$p(m_k|y_i, y_j, X) = \frac{\exp(A_m(X, y_i, y_j)[k])}{\sum_{k'=1}^{N_m} \exp(A_m(X, y_i, y_j)[k'])}. \quad (5)$$

Thus, we define the energy for the mark m as (see Appendix A.2 for details):

$$V_m(y_m|y_i, y_j, X) = -A_m(X, y_i, y_j)[k_{y_m}] + \ln\left(\sum_{k=1}^{N_m} \exp(A_m(X, y_i, y_j)[k])\right) \quad (6)$$

133 with k_{y_m} equal to the mark value y_m mapped to $[0, N_m]$; $k_{y_m} = N_m \frac{y_m - m_{\text{min}}}{m_{\text{max}} - m_{\text{min}}}$.

134 In practice, we infer a tensor $A_m(X)$ of shape (H, W, N_m) for an image of
 135 shape $(H, W, 3)$, then extract the values of $A_m(X)$ at coordinates (y_i, y_j, k_{y_m}) ,
 136 performing bilinear interpolation.

137 3.1.3. FCN for small objects in large images

138 For our application, we need a Fully Convolutional Network (FCN) [25]
 139 capable of responding to small objects in large images.

140 A straightforward approach to learning a position probability map $A_{\text{pos}}(X)$
 141 would be to directly infer a heatmap of centers: i.e., obtained by applying a
 142 Gaussian blur on a binary map of object centers (Figure 4b). However, these
 143 maps are sparse (non-zeros probabilities are under-represented), and at inference
 144 we observe some connectivity added in between blobs, making the separation of
 145 instances more difficult (see Figure 4b).

We circumvent this problem by learning a proxy-task: The model is first
 trained to infer a map of 2D unit-length vectors $A_{\text{vec}}(X)$ that point towards
 the closest instance center (Figure 4c) similarly to [26]. We then apply the
 divergence operator, as object centers now correspond to convergence points in
 this vector field [27] (Figure 4d):

$$A_{\text{pos}}(X) = a \cdot \text{div}(A_{\text{vec}}(X)) + b. \quad (7)$$

146 The backbone architecture corresponds to a Unet [23], which outputs for
 147 an image X a tensor $F(X)$ with N_F features per pixel. From that feature
 148 representation are extracted $A_{\text{pos}}(X)$ and the $A_m(X) \forall m \in \{a, b, \alpha\}$.

In our experiments we train the Unet to minimize the following cost function
 for position term [27] and mark terms:

$$\mathcal{L}_{\text{pos}}(X, Y^{\text{GT}}) = \text{MSE}(A_{\text{vec}}(X), A_{\text{vec}}(Y^{\text{GT}})) + \text{BCE}(A_{\text{pos}}(X), A_{\text{pos}}(Y^{\text{GT}})), \quad (8)$$

$$\mathcal{L}_m(X, Y^{\text{GT}}) = \frac{1}{|P|} \sum_{p \in P} \text{CE}(\text{Softmax}(A_m(X, p_i, p_j)), A_m(Y^{\text{GT}}, p_i, p_j)), \quad (9)$$

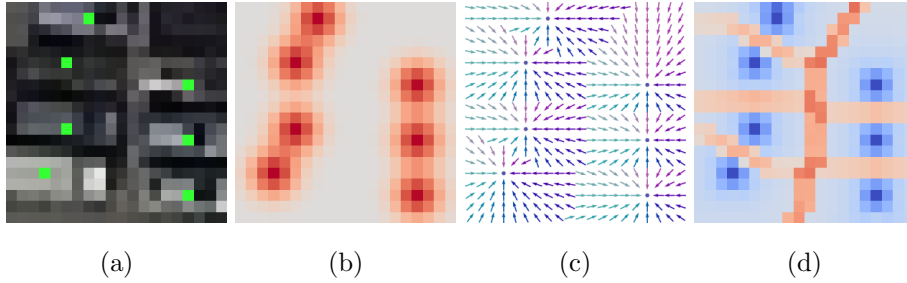


Figure 4: Closeup image with centers overlay (a), centers heatmap (b), vector field (c) and divergence (d) (red > 0 , blue < 0)

149 with MSE the Mean Squared Error, BCE the Binary Cross Entropy and CE the
 150 Cross Entropy. The tensors $A_{\text{vec}}(Y^{\text{GT}})$, $A_{\text{pos}}(Y^{\text{GT}})$ and $A_m(Y^{\text{GT}}, p_i, p_j)$ are
 151 respectively the vector, position, and mark tensors build from the annotated
 152 Ground Truth.

153 3.2. Energy priors

154 The total energy model encompasses several priors as energies, those allow
 155 to regularize the configuration against the data terms.

156 3.2.1. Object priors

These are functions of the current point y only. For $k = \text{ratio, area}$:

$$V_k(y) = -\exp\left(-0.5(f_k(y) - \mu_k)^2 \sigma_k^{-2}\right) \quad (10)$$

157 with $f_{\text{ratio}}(y) = y_b/y_a$ and $f_{\text{area}}(y) = y_a y_b$, and μ_k, σ_k being respectively the
 158 target value and the standard deviation.

159 3.2.2. Interaction priors

The following priors depend on the neighborhood of the point y . The term in Eq. 11 penalizes overlapping objects (with t_{overlap} the overlap threshold), Eq. 12 favors aligned objects ($t_\alpha = 0$ favors parallel objects, while $t_\alpha = \pi/2$ prefers perpendicular objects), Eq. 13 and 14 are respectively repulsive and attractive priors, finally Eq. 15 allows to adjust the energy of neighborless points. Some

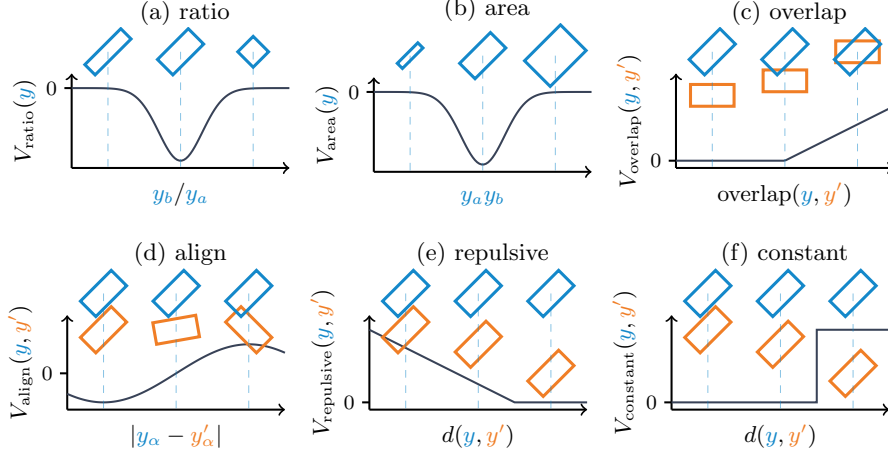


Figure 5: Illustration of some energy priors: (a) and (b) Eq. 10, (c) Eq. 11, (d) Eq. 12, (e) Eq. 13, (f) Eq. 15.

of these priors are illustrated in Figure 5.

$$V_{\text{overlap}}(y|\mathcal{N}_y) = \max_{y' \in \mathcal{N}_y} \left\{ \text{Relu} \left(\frac{\text{area}(y' \cap y)}{\min\{\text{area}(y'), \text{area}(y)\}} - t_{\text{overlap}} \right) \right\} \quad (11)$$

$$V_{\text{align}}(y|\mathcal{N}_y) = \min_{y' \in \mathcal{N}_y} \{-\cos(|y_\alpha - y'_\alpha| - t_\alpha)\} \quad (12)$$

$$V_{\text{repulsive}}(y|\mathcal{N}_y) = \max_{y' \in \mathcal{N}_y} \left\{ 1 - \frac{d(y, y')}{d_{\text{max}}} \right\} \quad (13)$$

$$V_{\text{attractive}}(y|\mathcal{N}_y) = \min_{y' \in \mathcal{N}_y} \left\{ \frac{d(y, y')}{d_{\text{max}}} \right\} \quad (14)$$

$$V_{\text{constant}}(y|\mathcal{N}_y) = \mathbb{1}_{|\mathcal{N}_y|=0} \quad (15)$$

160 where $\text{Relu}(x) = \max(0, x)$ for $x \in \mathbb{R}$.

161 3.2.3. Learning prior parameters

162 While the parameters of the priors can be set manually (or by trial and
 163 error), we can fine tune the parameters of those within the θ inference procedure
 164 detailed in Section 4.2.

Eq. 12 can be reformulated as:

$$V_{\text{ratio}}(y|\theta) = -\exp\left(-0.5(f_{\text{ratio}}(y) - \theta_{\mu, \text{ratio}})^2 \theta_{\sigma, \text{ratio}}^{-2}\right) \quad (16)$$

and similarly for *area*. The other priors can be parametrized as such, giving:

$$V_{\text{overlap}}(y|\mathcal{N}_y, \theta) = \max_{y' \in \mathcal{N}_y} \left\{ \text{Relu} \left(\frac{\text{area}(y' \cap y)}{\min\{\text{area}(y'), \text{area}(y)\}} - \theta_{t, \text{overlap}} \right) \right\} \quad (17)$$

$$V_{\text{repulsive}}(y|\mathcal{N}_y, \theta) = \max_{y' \in \mathcal{N}_y} \left\{ \text{Relu} \left(1 - \frac{d(y, y')}{d_{\text{max}}} - \theta_{t, \text{repulsive}} \right) \right\} \quad (18)$$

$$V_{\text{attractive}}(y|\mathcal{N}_y, \theta) = \min_{y' \in \mathcal{N}_y} \left\{ \text{Relu} \left(\frac{d(y, y')}{d_{\text{max}}} - \theta_{t, \text{attractive}} \right) \right\}. \quad (19)$$

165 4. Learning model parameters

166 As introduced in Eq. 3, the energy of a configuration Y will depend on the
 167 energy term weights θ_w . The weights θ_w encode the relative importance of the
 168 data and prior terms in the point process density. These model parameters can
 169 be set manually (i.e., by trial and error), or learned on available training data.

More generally some energy terms can also be parametrized by θ such that
 energy terms are also function of θ , $V_k(y|X, \mathcal{N}_y, \theta)$ (see Section 4.2.4). Regard-
 less of the method, the intuition is to select a $\hat{\theta}$ that approximates θ^* such that:

$$Y^{\text{GT}} = \underset{Y \in \mathcal{Y}}{\text{argmin}} U(Y|X, \theta^*), \quad (20)$$

170 i.e., the configuration that minimizes $Y \mapsto U(Y|X, \theta^*)$ is the Ground Truth
 171 (GT) configuration Y^{GT} corresponding to image X .

172 We first present the previous method based on local perturbation and linear
 173 programming to learn θ_w . We then elaborate on how we adapt maximum like-
 174 lihood learning for Energy Based Models (EBM) to point processes for learning
 175 the parameters θ .

176 4.1. Learning energy weights with local perturbations

177 Craciun et al. [18] and Yu et al. [22] uses random perturbations on the
 178 Ground Truth Y^{GT} to generate non-valid configurations and build a set of linear
 179 constraints to estimate θ_w .

As a matter of fact, re-formulating Eq. 20 as a local constraint yields:

$$U(Y^+, X, \theta^*) < U(Y^-, X, \theta^*), \quad Y^- \sim Q(Y^+ \rightarrow \cdot) \quad (21)$$

180 where Y^- corresponds to a configuration obtained by passing the Ground Truth
 181 Y^{GT} through a perturbation kernel Q . The valid configuration Y^+ is either set
 182 as the Ground Truth Y^{GT} , or Y^{GT} with some light perturbation (e.g., displacing
 183 the points by a Gaussian random variable of limited variance).

184 Given a training set \mathcal{S} of image and valid configuration pairs (X_l, Y_l^+) , $l =$
 185 $0, \dots, |\mathcal{S}|$, we can build linear constraints as in Eq. 21 from this training set.
 186 While Craciun et al. [18] and Yu et al. [22] use linear programming to find θ_w ,
 187 this poses a limit on the number of constraints we can account for, in order to
 188 avoid an over-constraining of the problem. Moreover, when the data labeling is
 189 imperfect or noisy, constraints can contradict each other and make the problem
 190 unsolvable. Finally, this approach only allows to learn the weights parameters
 191 θ_w , thus prior parameters such as $\theta_{t,overlap}$, in Eq. 17 have to be set manually
 192 or with a separate optimization procedure.

193 4.2. Maximum likelihood learning for Energy Based Models

194 4.2.1. Maximizing likelihood

LeCun et al. [20] proposes to learn Energy Based Models (EBM) to maxi-
 mize the likelihood for the data \mathcal{S} :

$$p\left(Y_1^+, \dots, Y_{|\mathcal{S}|}^+ | X_1, \dots, X_{|\mathcal{S}|}, \theta\right) = \prod_{k=1}^{|\mathcal{S}|} p\left(Y_k^+ | X_k, \theta\right) \quad (22)$$

195 Here we denote valid configurations as Y^+ , that can be defined as the Ground
 196 Truth Y^{GT} , or as the ground truth plus a small perturbation (in the case where
 197 the Ground Truth is known to be imperfect) $Y^+ \sim Y^{\text{GT}} + \mathcal{N}(0, \sigma_+)$.

The negative log likelihood is then given, for parameters θ_n at step n , as:

$$\mathcal{L}_{nll}(\theta_n, \mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{k=1}^{|\mathcal{S}|} \left(U(Y_k^+ | X_k^+, \theta_n) + \frac{1}{\beta} \log \int_{Y \in \mathcal{Y}} \exp(-\beta U(Y | X_k, \theta_n)) \right) \quad (23)$$

with β an inverse temperature parameter (from the Gibbs distribution), that
 has no effect on the position of the minimum [20]. The gradient of \mathcal{L}_{nll} on θ_n
 is shown to be

$$\frac{\partial \mathcal{L}_{nll}(\theta_n, Y_i^+, X_i)}{\partial \theta_n} = \frac{\partial U(Y_i^+ | X_i, \theta_n)}{\partial \theta_n} - \int_{Y \in \mathcal{Y}} \frac{\partial U(Y | X_i, \theta_n)}{\partial \theta_n} p(Y | X_i, \theta_n) \quad (24)$$

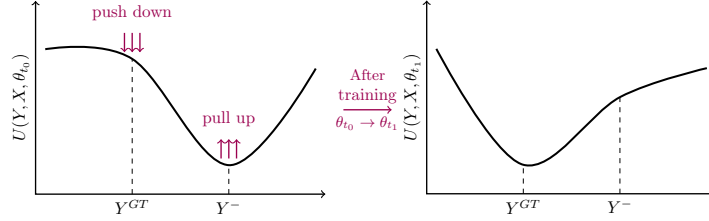


Figure 6: Effect of training the energy with contrastive samples generated from the current θ_n (figure adapted from [20]).

198 where $p(Y|X_i, \theta_n) = \exp(-\beta U(Y|X_i, \theta_n)) / \int_{\tilde{Y} \in \mathcal{Y}} \exp(-\beta U(\tilde{Y}|X_i, \theta_n))$.

199 While the integral $\int_{Y \in \mathcal{Y}} \frac{\partial U(Y, X_i, \theta_n)}{\partial \theta_n} p(Y|X_i, \theta_n)$ remains intractable, it can
 200 be approximated through Monte Carlo sampling, where $\tilde{Y}^0, \dots, \tilde{Y}^N$ are drawn
 201 from the law defined by $p(\cdot|X_i, \theta_n)$, giving

$$\int_{Y \in \mathcal{Y}} \frac{\partial U(Y, X_i, \theta_n)}{\partial \theta_n} p(Y|X_i, \theta_n) \simeq \frac{1}{N} \sum_{k=1}^N \frac{\partial U(\tilde{Y}^k, X_i, \theta_n)}{\partial \theta_n} \quad (25)$$

202 4.2.2. Contrastive divergence

203 Hinton [28, 29] proposes to use a single sample in his *contrastive divergence*
 204 method. This method also uses few simulation steps over a Monte Carlo Markov
 205 chain (MCMC), starting from the desired answer Y^{GT} .

206 The general idea is to generate *contrastive samples* Y^- that follow the den-
 207 sity derived from $U(\cdot, X, \theta_n)$ at step t of the optimization. Then we proceed to
 208 update θ_n to θ_{n+1} , by gradient descent, with the goal to minimize the energy
 209 of the *valid sample* Y^+ , while maximizing the energy of the *contrastive sample*
 210 Y^- (see Figure 6).

211 While the local perturbation method (Section 4.1) only learns from con-
 212 straints with states in the neighborhood of the Ground Truth Y^{GT} , the con-
 213 trastive divergence method allows learning by contrast against any state in \mathcal{Y} . It
 214 also focuses the contrastive samples on low energy states (instead of uniformly
 215 sampling in \mathcal{Y} which would be inefficient).

216 Du [21] proposes an adaptation of Hinton’s method [28, 29], making use of
 217 a replay buffer. The replay buffer saves Markov chain results for initialization

218 in the next optimization steps in order to save computing time. This allows
 219 reducing the long simulation time necessary to pass the burn-in period of the
 220 Markov Chain and get samples Y^- , by picking an initial state with a known
 221 low energy $U(\cdot, X, \theta_n)$.

222 For a Markov chain initialized as \tilde{Y}_0 , simulated for K steps, we take the final
 223 configuration \tilde{Y}_K as our contrastive sample Y^- . The replay buffer makes the
 224 initial state at step t , \tilde{Y}_0 be chosen as the resulting configuration Y^- from last
 225 step $t-1$, with probability 0.99, or else a random configuration chosen uniformly
 226 in \mathcal{Y} .

In Algorithm 1 we adapt the method from Du et al. [21] to our approach,
 minimizing the loss \mathcal{L} using the Stochastic Gradient Descent (SGD) method
 [30];

$$\begin{aligned} \mathcal{L}(\theta, Y^+, Y^-, X) &= U(Y^+, X, \theta) - U(Y^-, X, \theta) + \gamma \mathcal{R} \\ \mathcal{R} &= \sum_{Y=Y^+, Y^-} \frac{1}{|Y|} \sum_{y \in Y} V(y|X, \mathcal{N}_y^Y, \theta) \end{aligned} \quad (26)$$

227 with γ the weight of regularization term \mathcal{R} . We introduce the regularization
 228 term to avoid an explosion of the per-point energy $V(y|X, \mathcal{N}_y^Y, \theta)$.

Algorithm 1 Maximum likelihood learning

- 1: $\beta \leftarrow \emptyset, n \leftarrow 0$
 - 2: **while** not converged **do**
 - 3: pick X, Y^{GT} from \mathcal{S}
 - 4: $\tilde{Y}_0 \sim \beta_X$ with probability 0.99, else $\mathcal{U}(\mathcal{Y})$
 - 5: $Y^- \leftarrow \text{SampleMPP}(Y_0, X, \theta_n)$ ▷ Defined in Algorithm 2
 - 6: $\beta_X \leftarrow Y^-$
 - 7: $\Delta\theta_n \leftarrow \nabla_{\theta_n} \mathcal{L}(\theta_n, Y^+, Y^-, X)$
 - 8: Update θ_{n+1} with $\Delta\theta_n$ using SGD
 - 9: $n \leftarrow n + 1$
 - 10: **end while**
-

229 *4.2.3. Relating contrastive divergence to manual method*

230 In practice, the contrastive divergence method relates to the empirical trial
231 and error procedure used in previous works to set θ . Indeed, oftentimes the
232 manual procedure to find weights θ_w goes as follows:

- 233 1. the user sets some approximate values to weights of $\tilde{\theta}_w$,
- 234 2. the user simulates configurations $Y \sim U(\cdot|X, \tilde{\theta})$,
- 235 3. if the inference is of good quality, the procedure ends. Otherwise, the
236 user updates values for weights $\tilde{\theta}_w$, picked to counteract the observed bad
237 configurations; (e.g., *the configuration has too much overlap* \rightarrow *increase*
238 *the weight of the overlap energy*). Then go back to step 2.

239 In a broad sense, the repeat of steps 2 and 3 maps to the alternated sampling
240 (line 5) and loss minimization (line 7) in Algorithm 1. As step 3 in the manual
241 procedure, attempts to maximize the energy of the (bad) sampled configuration,
242 minimizing the loss in Algorithm 1 tends to increase $U(Y^-, X, \theta)$ and decrease
243 $U(Y^+, X, \theta)$.

244 *4.2.4. Learning more than just weights*

245 Previous works [18, 22] used learning methods to infer the weights of the
246 energy terms, corresponding to θ_w in our model. Given the formulation of the
247 loss in Eq. 26, and the use of gradient descent as shown in Algorithm 1, the
248 learning task can be extended to any parameter of the model as long as the
249 gradient can be back-propagated to it. Machine learning frameworks such as
250 PyTorch [31] allow for fast automatic back-propagation of gradients.

251 In section 6 we show results with different depth of learned parameters:
252 First using contrastive divergence only to infer energy term weights θ_w (10
253 parameters), then in a second model, inferring both θ_w and the data and prior
254 energies parameters $\theta_{t,\text{pos}}, \theta_{t,\text{overlap}}, \dots$ (18 parameters, see in Section 3.2.3).

255 It is possible to infer the parameters of the whole data energy model (i.e., the
256 backbone CNN), however this proves quite computationally expensive; Training
257 the CNN requires many epochs as parameters are numerous ($> 10^6$ parameters).

258 Meanwhile, the contrastive divergence method (Algorithm 1) requires simulating
 259 a few Markov Chain steps in between epochs to generate contrastive samples,
 260 which increases the computational complexity of each epoch.

When inferring the prior parameters with contrastive divergence, we estimate prior energy terms (functions that map a configuration/image pair to a scalar) that maximize the likelihood on the training data. However, the available functions are constrained by the form of the priors set in Eq. 16 to 19. As these formulations might not necessarily allow to access the optimal function, we propose a more generic interaction prior formulation:

$$V_{\text{inter.}}(y, \mathcal{N}_y, \theta) = \sum_{y' \in \mathcal{N}_y} w_{y,y'} v_{y,y'}, \quad (27)$$

261 where point to point interaction energies $v_{y,y'}$ are weighted by $w_{y,y'}$ (with
 262 $\sum_{y'} w_{y,y'} = 1$, acting as an aggregation operator; aggregating the $|\mathcal{N}_y|$ values
 263 $v_{y,y'}$ into a single value for each y). Formulating v and w as generic function of
 264 y and y' parametrized by θ (e.g., a lightweight Multi-Layer Perceptron (MLP),
 265 with > 100 parameters), would allow inferring an interaction model within a
 266 wider space of functions. This proposed approach might be the subject of future
 267 publications.

268 5. Sampling configurations from the energy model

Given an image X and estimated model parameters $\hat{\theta}$, $U(Y|X, \hat{\theta})$ defines the density of the point process (Eq. 1). In this section we develop on the method used for sampling from this probability law. Moreover, this sampling method allows inferring \hat{Y} , that estimate Y^* defined as

$$Y^* = \operatorname{argmin}_{Y \in \mathcal{Y}} U(Y|X, \hat{\theta}). \quad (28)$$

269 This is achieved by sampling the law derived from $U(Y|X, \hat{\theta})/T$, with T de-
 270 creasing geometrically (see Section 5.2).

271 *5.1. Reversible Jump Monte Carlo Markov Chain*

272 In order to simulate the PP given by its energy, we rely on the Reversible
 273 Jump Monte Carlo Markov Chain (RJMCMC) [32] algorithm as in [15, 16,
 274 17, 18]. This method extends the Metropolis Hastings algorithm allowing to
 275 explore a state space of varying dimension. Indeed, one needs to be able to
 276 explore all possible configurations $Y \in \mathcal{Y}$, with any number of points. Thus, the
 277 dimensionality — proportional to the number of points — is not fixed through
 278 the optimization process. The convergence of this algorithm is ensured — with
 279 enough steps — as long as a uniform birth-death kernel in $S \times M$ is implemented.

280 To speed up convergence Green [32] uses translation/rotation/scaling ker-
 281 nels. Moreover, we add a non-uniform birth kernel that proposes points based
 282 on a non-uniform density [16]. This proposal density is referred as a birth map
 283 (Section 5.1.2).

The RJMCMC relies on an accept-reject mechanism, where the new state Y' , proposed by the kernel Q from current state Y , is accepted with probability r [33]:

$$r(Y \rightarrow Y') = \min \left(1, \frac{Q(Y' \rightarrow Y)}{Q(Y \rightarrow Y')} \exp \left(-\frac{U(Y'|X, \theta) - U(Y|X, \theta)}{T} \right) \right), \quad (29)$$

284 with $T \in \mathbb{R}$ the temperature, so that we simulate $U(Y|X, \theta)/T$.

285 The kernel Q is composed of several sub-kernels, two of which are the birth
 286 and death kernels. At each Markov Chain step, a kernel Q_k is picked with
 287 probability p_k before proposing the new state Y' .

288 *5.1.1. Birth and Death kernels*

289 The essential Birth-Death kernel Q_{BD} is defined as follows:

290 *Birth.* With probability p_B , add one point y sampled uniformly in $S \times M$. Then,
 291 with $Y' = Y \cup \{y\}$,

$$\frac{Q(Y' \rightarrow Y)}{Q(Y \rightarrow Y')} = \frac{p_D Q_D(Y' \rightarrow Y)}{p_B Q_B(Y \rightarrow Y')} = \frac{p_D}{p_B} \frac{\lambda |S|}{|Y| + 1} \quad (30)$$

292 where $|S|$ is a measure of space S , and λ the intensity of the point process.

293 *Death.* With probability p_D , remove one point $y \in Y$ chosen uniformly amongst
 294 all points in Y . Then, with $Y' = Y \setminus \{y\}$,

$$\frac{Q(Y' \rightarrow Y)}{Q(Y \rightarrow Y')} = \frac{p_B Q_B(Y' \rightarrow Y)}{p_D Q_D(Y \rightarrow Y')} = \frac{p_B |Y|}{p_D \lambda |S|} \quad (31)$$

295 For our experiments we will set $p_B = p_D$. The value of p_B and p_D will depend
 296 on the other kernel types added to the method so that $\sum_{k \in \text{kernel types}} p_k = 1$.

297 5.1.2. Birth maps

298 While the uniform birth-death kernel allows exploring the whole space $S \times M$,
 299 it is more efficient (i.e., yields a higher acceptance rate) to propose points with
 300 a more adapted density [16].

A birth-death kernel ($Y' = Y \cup \{y\}$) with proposal density $y \sim d(y)$, gives:

$$\frac{Q(Y' \rightarrow Y)}{Q(Y \rightarrow Y')} = \frac{p_D Q_D(Y' \rightarrow Y)}{p_B Q_B(Y \rightarrow Y')} = \frac{\lambda}{d(y)(|Y| + 1)} \quad (32)$$

We are able to compute a discrete distribution over the discretized image and mark space P^3 thanks to the position energy term V_{pos} (Eq. 4) and marks energy terms V_m (Eq. 6) being derived from tensors $A_{\text{pos}}(X)$ (Eq. 7) and $A_m(X)$. To sample on $S \times M$, we sample a pixel q in discrete space P , then y is picked uniformly inside the pixel q . With $|q| = 1$ is the measure of one pixel, the density $d(y)$ becomes:

$$d(y) = \frac{d(q)}{|q|}. \quad (33)$$

We define the sampling density, by only considering the data energy change brought by the new point, regardless of interactions with points in Y :

$$d(q) = \frac{1}{Z_d} \exp \left(- \frac{\theta_{w,\text{pos}} V_{\text{pos}}(q_i, q_j | X) + \sum_{m \in \{a,b,\alpha\}} \theta_{w,m} V_m(q_m | q_i, q_j, X)}{T} \right) \quad (34)$$

301 Since V_{pos} and V_m are derived from tensors $A_{\text{pos}}(X)$ and $A_m(X)$, we can
 302 compute the normalizing constant Z_d over the discrete space of pixel positions⁴.

³ P is the discretized space $S \times M$, defined as $P = \llbracket 1, H \rrbracket \times \llbracket 1, W \rrbracket \times \prod_{m \in \{a,b,\alpha\}} \{m_k, k = 1, \dots, N_m\}$.

⁴The conditioning of each mark to the position also allows to normalize each mark density

303 *5.2. Simulated annealing*

304 As such, the RJMCMC simulates a sequence of states $(Y_t)_{t=0,\dots,K}$ which
305 distribution density at equilibrium is proportional to $\exp(-U(Y|X,\theta)/T)$. To
306 estimate configuration Y^* that minimizes the energy U , for image X and pa-
307 rameters \hat{Y} (i.e., infer \hat{Y}), we use simulated annealing [34]; The temperature T_t
308 decreases geometrically as we simulate the density $\exp(-U(Y|X,\hat{\theta})/T_t)$. While
309 convergence is guarantee for a logarithmic decrease, in practice it involves too
310 long simulation times. We use $T_{t+1} = \alpha T_t$ with $\alpha = 0.997$.

311 *5.3. Parallelization*

312 In its canonical implementation, the RJMCMC algorithm operates sequen-
313 tially over the points in the image — It may only add/remove/transform one
314 point at a time —, which increases simulation time linearly with the number of
315 objects. With parallelization, we aim to take advantage of the spatial Marko-
316 vianity of the point process; i.e., points further that a distance d_{\max} have no
317 effect on each other’s energy. Depending on the energy model (from which d_{\max}
318 is derived), the Green ratio (acceptance probability of a perturbation, in Eq.
319 29) of two kernel perturbations distant enough in space, is unchanged whether
320 these perturbations are done in any order [35]; Thus, both perturbations can be
321 computed and executed in parallel with no change on the resulting distribution.

322 As in Verdié et al. [35], we split the image into square cells of size $|c|$ that
323 are each assigned one set s (each set corresponding to one color in Figure 7).
324 These sets are referred as mic-sets in [35], for Mutually Independent Cells. Set
325 size $|c|$ is chosen so that moves in cells of one set (or color) have independent
326 acceptance ratios: i.e., those moves can be performed in any sequential order
327 —thus in parallel— without any change in the ratios of acceptance.

328 We show that our energy formulation requires cells of size $2d_{max} + 2\delta_{\max}$
329 (see Appendix B) where d_{max} is the maximum interaction distance and δ_{\max}
330 the maximum translation distance for a perturbation.

independently.

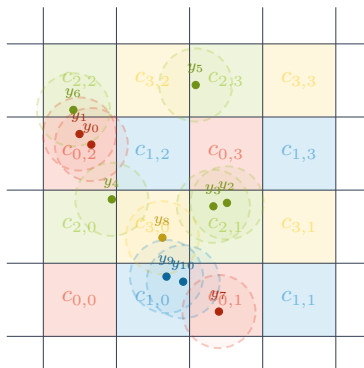


Figure 7: Space partitioning, where each color corresponds to a different mic-set. For cell $c_{s,z}$, is the z^{th} cell of mic-set s . For every point y we display its interaction radius d_{max} as a semi-transparent dotted circle.

331 Contrary to [35], we do not use a quad tree structure of cells, but rather
 332 a regular grid of $|c|$ sized cells. To achieve an efficient sampling of cells (i.e.,
 333 avoiding spending time in cells with no evidence of objects), weights are assigned
 334 to each cell to focus the sampling on the relevant ones. The sampling procedure
 335 goes as follows:

- 336 1. pick a kernel Q_k with probability p_k ,
- 337 2. pick one mic-set s with probability $p(s)$,
- 338 3. keep each cell in mic-set s with probability $p(c|s)$
- 339 4. for every cell c in set s run kernel Q_k restricted to c (denoted Q_k^c)

The process of cell selection (step 3), allows to limit the number of cells processed at once on big images, hence limiting the computational cost and maintaining the desired sampling density; For a birth kernel Q_B^c , the kernel density over the whole image (Eq. 29) is

$$Q(Y \rightarrow Y') = p_B p(s) p(c|s) Q_B^c(Y \rightarrow Y'). \quad (35)$$

Kernel Q_B^c , samples new points in cell c with density $d_c(q) = d(q) / \sum_{q' \in c} d(q')$ (i.e., density d normalized on cell c). To ensure the new sampling density

$p(s)p(c|s)d_c(q)$ corresponds to $d(q)$, we set:

$$p(c|s) = \frac{d(c)}{\sum_{c' \in s} d(c')}, \quad p(s) = d(s), \quad (36)$$

340 with $d(c) = \sum_{q' \in c} d(q')$, $d(s) = \sum_{c \in s} d(c)$. In practice, skewing probability
 341 $p(c|s)$ to ensure more cells are kept for simulation (e.g., replacing $p(c|s)$ by
 342 $p'(c|s) = \min(1, np(c|s))$, with $n > 1$), will increase the amount of parallel
 343 computations of cells (thus reducing simulation time) at the cost of deteriorating
 344 the approximation of sampling density $d(p)$.

345 In Algorithm 2 and later sections, we denote the points in configuration Y
 346 that fall in cell c as $C_c(Y)$.

347 5.4. Jump diffusion

Instead of random perturbations, we can use the energy gradient to explore the space more efficiently. This is the idea behind stochastic diffusion (or Langevin) dynamics [22, 36, 37]. If Y_t and T_t denote respectively the configuration and temperature at time t , then the configuration for the next step of the Markov chain is given by $Y_{t+1} = Y_t + dY_t$, with step size δ :

$$dY_t = -\delta \frac{\partial U(\hat{\theta}, X, Y_t)}{\partial Y_t} + dw_t \sqrt{2T_t}, \quad dw_t \sim \mathcal{N}(0, \delta). \quad (37)$$

348 At high temperature, the Brownian motion from dw_t ensures an exploration
 349 of the space. At low temperature the Brownian motion is negligible and the
 350 diffusion performs as a gradient descent. The diffusion alone allows to explore
 351 the space \mathcal{Y} locally around Y_t at a fixed dimension (d , fixed number of points).
 352 To explore (or *jump*) across dimensions we make use of the birth and death
 353 kernel as defined previously.

354 In practice, as parallelization requires some maximum displacement on points
 355 (see Section 5.3), the step dY_t from Y_t to Y_{t+1} is clipped; For each point
 356 $y \in Y_t$ updated to y' , we bound the i and j step components by δ_{\max} (i.e.,
 357 $|y_i - y'_i| \leq \delta_{\max}$, and similarly for j).

358 The resulting sampling method is outlined in Algorithm 2.

Algorithm 2 SampleMPP($Y_0, X, \theta, N_{\text{smp1}}$)

```
1: for  $t = 0, \dots, N_{\text{smp1}}$  do
2:   Pick diffusion with probability 0.8, else jump
3:   Pick one mic-set  $s$  with probability  $p(s)$ 
4:   Keep each  $c$  in  $s$  with probability  $p(c|s)$  to make  $\tilde{s}$ 
5:    $Y_{t+1} \leftarrow Y_t$  ▷ unvisited cells keep previous state
6:   for all  $c \in \tilde{s}$  do
7:     if diffusion then
8:        $dw \sim \mathcal{N}(0, \delta)$ 
9:        $C_c(Y_{t+1}) \leftarrow C_c(Y_t) - \delta \nabla_{C_c(Y_t)} U(Y_t|X, \theta) + dw\sqrt{2T_t}$ 
10:    else
11:       $Y' \sim Q_{BD}^c(Y_t \rightarrow \cdot)$  ▷ perform move in  $c$ 
12:       $r \leftarrow \frac{Q(Y_t' \rightarrow Y_t)}{Q(Y_t \rightarrow Y')}$   $\exp\left(-\frac{\Delta U(Y_t \rightarrow Y'|X, \theta)}{T_t}\right)$ 
13:       $C_c(Y_{t+1}) \leftarrow C_c(Y')$  with probability  $\min(1, r)$ 
14:    end if
15:  end for
16:   $T_{t+1} \leftarrow \alpha T_t$ 
17: end for
```

359 6. Application

360 6.1. Datasets

361 Our application goal is the detection of small objects in images from satellites
362 such as Pléiades⁵ or CO3D⁶, which have a typical spatial resolution (or ground
363 sampling distance) of 0.5 meter per pixel.

364 To train the CNN backbone, and infer the model parameters $\hat{\theta}$, we compile
365 DOTA_{gsd0.5}, a 0.5 meter per pixel vehicles in remote sensing dataset, from the
366 DOTA dataset [38]. This dataset is built by sub-sampling images from DOTA
367 to the desired spatial resolution.

368 To test the resilience of our models against noise, we test the methods on

⁵Constellation Pléiades [pleiades.cnes.fr]

⁶Constellation Optique 3D [intelligence-airbusds.com]

369 the same data adding a Gaussian noise (with $\sigma = 0.3$, see Figure 9).

370 Moreover, we evaluate models on data provided by Airbus Defense and Space
371 (ADS), this aerial data is sub-sampled to the desired resolution, emulating the
372 satellites characteristics above-mentioned. This dataset is unlabeled, thus will
373 only serve to evaluate the models qualitatively, and was not used for training.

374 6.2. Comparison to PP based on contrast measures

375 To demonstrate the need for learned likelihood measures for our data, we
376 evaluate a few contrast measures found in literature [16, 19]. Table 1 formulates
377 these measures by denoting μ_y, σ_y^2 the empirical mean and variance of pixels
378 inside shape y ; μ_s, σ_s^2 mean and variance on shape contour s ; n_y the normal
379 vector of contour s , and ∇X the image gradient.

380 To assert the viability of such measures for this data, we compute a score
381 $s(y) = \exp(-V_c(y))$ for proposed detections y (both Ground Truth and ran-
382 domly scattered rectangles). The true objects represent 1% of this experiment’s
383 dataset⁷. A well suited measure $s(y)$ should allow classifying easily between
384 Ground Truth objects and random, non-valid, objects. We plot precision-recall
385 curves for several of those contrast measures in Figure 8 both for a sample im-
386 age of DOTA_{gsd0.5}, and for a synthetic highly contrasted image. The Average
387 Precision (AP) values are reported in Table 1. We compare with the proposed
388 detection energy given by $V_p + V_m$.

389 The proposed measure, based on CNN output, outperforms classical contrast
390 based measures, as the CNN has learned to differentiate between relevant and
391 irrelevant contrasted objects⁸.

⁷A higher proportion of true objects would allow high average precision for useless estima-
tors predicting a constant value.

⁸Irrelevant objects include for instance AC units –a highly contrasted, car-sized object– or
some pavement features.

Measure	V_c formulation	AP_{real}	$AP_{\text{synthetic}}$
CNN output (ours)	$V_{\text{pos}}(y X) \sum_m V_m(y X)$	0.99	0.88
T-test [16]	$\frac{ \mu_y - \mu_s }{\sqrt{\frac{\sigma_y^2}{n_y} + \frac{\sigma_s^2}{n_s}}}$	0.13	0.99
Gradient [19]	$\int n_y(t) \cdot \frac{\nabla X(s(t))}{\sqrt{ \nabla X(s(t)) ^2 + \epsilon}} dt$	0.27	0.99
Constant	1.0	0.01	0.01

Table 1: Various contrast measures from literature and Average Precision (AP) computed on the DOTA image (AP_{real}) or synthetic image ($AP_{\text{synthetic}}$) from Figure 8

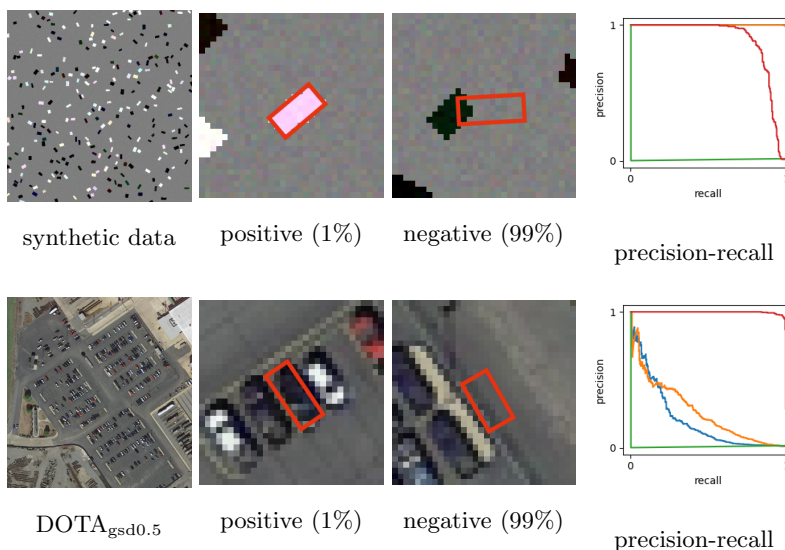


Figure 8: Comparing contrast measures on synthetic (first line) and real data (second line). Positive and negative samples, represent respectively 1% and 99% of the samples for which the metric is computed. For precision-recall; blue: T-test [16], orange: image-gradient[19], red to CNN output, green: constant value.

392 *6.3. Papangelou intensity as confidence score*

393 Classical CNNs for object detection yield a confidence score $s(y) \in \mathbb{R}$ for
 394 each proposed object y in the image. This confidence score is often interpreted,
 395 for each detection, as proportional to the probability of proposed element y to
 396 be a true (labeled) object, $s(y) \propto p(y|X)$. Applying a score threshold t_s gives a
 397 set of detections, for which metrics such as precision and recall can be computed.
 398 The detector performance over all possible threshold values is summed up via
 399 the Average Precision (AP); area under the precision-recall curve.

With the PP approach, the probability of one proposed point being an object
 of interest depends on the rest of the configuration \hat{Y} , $s(y|\hat{Y} \setminus \{y\}) \propto p(y|\hat{Y} \setminus$
 $\{y\}, X)$. This score is proportional to the Papangelou intensity [13], defined for
 every $y \in Y$ as:

$$\lambda(y|Y \setminus \{y\}) = \frac{h(Y|X, \hat{\theta})}{h(Y \setminus \{y\}|X, \hat{\theta})} = \exp\left(U(Y \setminus \{y\}|X, \hat{\theta}) - U(Y|X, \hat{\theta})\right), \quad (38)$$

400 The dependency of the score on the current configuration yields a complica-
 401 tion when computing the Average Precision: When applying a threshold t_s to
 402 prune the configuration Y into $Y' \subset Y$, for any $y \in Y'$, the score $s(y|Y \setminus \{y\})$
 403 may differ from $s(y|Y' \setminus \{y\})$. With a score of the form $s(y)$, that only depends
 404 on y and the image X , such as those from classical CNN models, the score from
 405 one object after pruning is unchanged.

In the PP case, we compute the scores by sequentially removing the lowest
 scoring point until there is none left; i.e., we build a sequence of configurations
 $Y_1 \supset Y_2 \dots Y_{|\hat{Y}|-1} \supset Y_{|\hat{Y}|} \supset \emptyset$, with $Y_1 = \hat{Y}$:

$$Y_{n+1} = Y_n \setminus \{y_n\}, y_n = \arg \min_{y \in Y_n} \lambda(y|Y_n \setminus \{y\}), \quad n = 1, \dots, |\hat{Y}| \quad (39)$$

406 This provides a pruning order $y_1, \dots, y_{|\hat{Y}|}$ of points in \hat{Y} . This ordering allows
 407 to plot the precision and recall curve⁹.

⁹Tracing the precision-recall curve only requires an ordering of the detected elements. Indeed, it only requires the sequence of $(\text{Recall}(t_s), \text{Precision}(t_s))$ pairs, which are obtained by sequentially pruning the lowest scoring points.

408 *6.4. Models*

409 In this article we show results on two CNN based models, and two PP models.

- 410 • CNN-local max.: Naive detection from the CNN backbone (used in the PP
411 models); we find objects through local maxima in the output probability
412 maps (or local minima in the energy maps).
- 413 • CNN-PP \blacklozenge : PP model with minimal inferred parameters: Pre-set priors
414 (i.e., with manually set parameters) as described in Sections 3.2.1 and
415 3.2.2, only the weight vector $\hat{\theta}_w$ is estimated from the training dataset.
- 416 • CNN-PP \blackstar : PP model with parametrized priors, as established in Section
417 3.2.3. The parameter vector $\hat{\theta}$ (which encompasses the weights $\hat{\theta}_w$ and
418 priors parameters) is estimated from the same training dataset.
- 419 • BBA-Vec. and YOLOV5-OBB: Lastly, we compare all of our models above
420 with BBA-Vec. from [8] and YOLOV5-OBB from [39, 40]. These models
421 are trained on the same data as the above-mentioned models.

422 *6.5. Results*

423 The above models are evaluated on a test split of the DOTA_{gsd0.5} dataset.
424 Metrics are computed by matching the object proposals and Ground Truth
425 using the IOU (Intersection Over Union), with a 0.25 threshold. We show
426 detection results in Figure 9, with a score threshold set for each model to the
427 one maximizing the F1 score. The Average Precision (AP) metrics (which are
428 threshold independent), are shown in Table 2.

429 We notice in Figure 9, that although Ground Truth labels are noisy —
430 which could have caused over constraining problem with linear programming
431 (see Section 4) — our models infer more regular configurations, that better fit
432 the objects and their physical constraints.

433 Testing the models on noisy data (Figure 9, line 4) and on the ADS data
434 (Figure 10), shows that the added priors allow the PP based models to be
435 more resilient to noise or areas of the image with limited information (e.g.,

Method	AP ₀	AP _N
BBA-Vec.	0.82	0.19
YOLOV5-OBB	0.86	0.10
CNN-local max.	0.86	0.55
CNN-PP [◆]	0.91	0.58
CNN-PP [★]	0.92	0.62

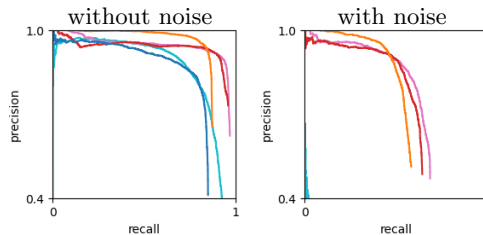


Table 2: Average Precision (AP) values, with AP₀ and AP_N, respectively AP for default and noisy data. Right : Precision Recall (PR) curves, color correspondence in table above.

436 low resolution, partial occlusions, shadows, see Figure 10, line 2). The ADS
 437 data inference results are obtained with the models trained on the DOTA_{gsd0.5}
 438 training set.

439 Lastly, CNN-PP[◆] and CNN-PP[★] show comparable performances (qualita-
 440 tively and quantitatively), demonstrating the capability of the contrastive diver-
 441 gence method to infer the prior/interaction model (via the prior parameters),
 442 thus limiting the trial and error inputs.

443 Our two PP based methods — running on an *Nvidia Quadro RTX 8000*
 444 Graphics Processing Unit — execute in 300s on average, for 16k parallel
 445 iterations (equivalent to 77k sequential iterations) for one image.

446 It is also possible to reduce the number of iterations of the point process
 447 sampler, at the cost of reduced precision.

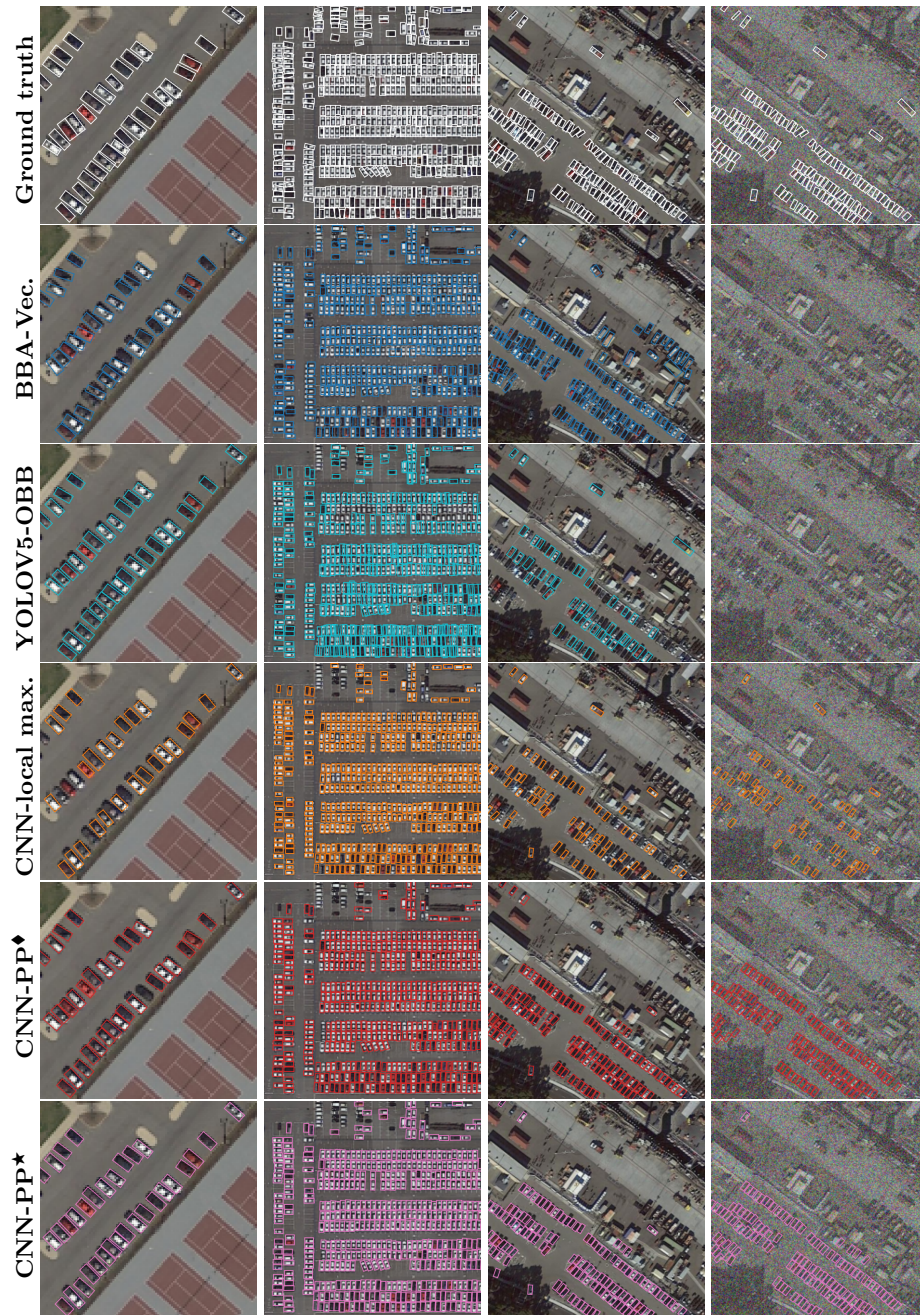


Figure 9: Samples of detection on the test dataset (columns 1-3), and the same dataset with additive Gaussian noise (columns 4). The score threshold (to not display low score objects) is set to maximize the $F1$ score for each model/dataset.



Figure 10: Model applied to ADS data. [© Airbus Defense and Space]

448 7. Conclusion and perspectives

449 The proposed method combines Convolutional Neural Networks — that ex-
450 tracts relevant information efficiently from images — with Point Processes —
451 which model interactions with few parameters — allowing to geometrically reg-
452 ularize the inferred configurations. It also improves resilience to input noise
453 compared to mainstream CNN models.

454 Moreover, we improve on previous PP models, by adapting Energy Based
455 Models (EBM) learning techniques to infer the parameters of the energy model,
456 alleviating manual trial and errors over parameters, and performing better than
457 manually set parameters. More importantly, our method allows inferring param-
458 eters on less precise labels and has fewer risks of over constraining the procedure
459 compared to linear programming. Lastly this method allows inferring deeper
460 parameters, contrary to only estimating prior term weights.

461 With the introduced scoring method based on the Papangelou intensity of
462 the PP, we show our models performs greatly in remote sensing images, espe-
463 cially when the images are noisy.

464 However, the use of Markov Chain based sampling for inference, necessarily
465 implies longer inference times compared to end-to-end neural network mod-
466 els. Inference time could be reduced by using approximations of the sampling
467 method as presented in Section 5, this could be investigated in future works.

468 The results shown in this paper, lead us to believe our models are able to
469 compensate for images with high noise. That is why our future works focus
470 on data such as Synthetic Aperture Radar (SAR) imagery. Moreover, we are
471 looking forward to applying our models to tasks with more prominent priors
472 in the likes of multiple object tracking (where object dynamics are crucial to
473 regularize the resulting tracks), building footprint or road network extraction
474 (where geometrical priors are strong).

475 **References**

- 476 [1] Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye, Object Detection in 20 Years: A
477 Survey, *Proc. IEEE* 111 (3) 257–276. doi:10.1109/JPROC.2023.3238524.
- 478 [2] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time
479 object detection with region proposal networks, *IEEE Trans. on Pattern*
480 *Anal. and Mach. Intell.* 39 (6) (2017) 1137–1149. doi:10.1109/TPAMI.
481 2016.2577031.
- 482 [3] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Uni-
483 fied, real-time object detection, in: *IEEE Conference on Computer Vision*
484 *and Pattern Recognition*, 2016, pp. 779–788. doi:10.1109/CVPR.2016.91.
- 485 [4] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense
486 object detection, *IEEE Trans. on Pattern Anal. and Mach. Intell.* 42 (2)
487 (2020) 318–327. doi:10.1109/TPAMI.2018.2858826.
- 488 [5] Y. Li, Y. Xing, Z. Wang, T. Xiao, Q. Song, W. Li, J. Wang, A framework
489 of maximum feature exploration oriented remote sensing object detection,
490 *IEEE Geosci. and Remote Sens. Lett.* 20 (2023) 1–5. doi:10.1109/LGRS.
491 2022.3228689.
- 492 [6] Y. Yao, G. Cheng, G. Wang, S. Li, P. Zhou, X. Xie, J. Han, On improving
493 bounding box representations for oriented object detection, *IEEE Trans.*
494 *on Geosci. and Remote Sens.* 61 (2023) 1–11. doi:10.1109/TGRS.2022.
495 3231340.
- 496 [7] T. Zhao, N. Liu, T. Celik, H.-C. Li, An arbitrary-oriented object detector
497 based on variant Gaussian label in remote sensing images, *IEEE Geosci.*
498 *and Remote Sens. Lett.* 19 (2022) 1–5. doi:10.1109/LGRS.2021.3087492.
- 499 [8] J. Yi, P. Wu, B. Liu, Q. Huang, H. Qu, D. Metaxas, Oriented object
500 detection in aerial images with Box Boundary-Aware Vectors, in: *IEEE*
501 *Winter Conference on Applications of Computer Vision*, 2021, pp. 2149–
502 2158. doi:10.1109/WACV48630.2021.00220.

- 503 [9] Y. Yang, X. Tang, Y.-M. Cheung, X. Zhang, F. Liu, J. Ma, L. Jiao,
504 AR2Det: An accurate and real-time rotational one-stage ship detector in
505 remote sensing images, *IEEE Trans. on Geosci. and Remote Sens.* 60 (2022)
506 1–14. doi:10.1109/TGRS.2021.3092433.
- 507 [10] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature* 521 (7553) (2015)
508 436–444. doi:10.1038/nature14539.
- 509 [11] S. He, H. Zou, Y. Wang, R. Li, F. Cheng, ShipSRDet: An end-to-end
510 remote sensing ship detector using super-resolved feature representation,
511 in: *IEEE International Geoscience and Remote Sensing Symposium*, 2021,
512 pp. 3541–3544. doi:10.1109/IGARSS47720.2021.9554079.
- 513 [12] R. LaLonde, D. Zhang, M. Shah, ClusterNet: Detecting small objects in
514 large scenes by exploiting spatio-temporal information, in: *IEEE/CVF*
515 *Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4003–
516 4012. doi:10.1109/CVPR.2018.00421.
- 517 [13] M. Van Lieshout, *Markov Point Processes and Their Applications*, Imperial
518 College Press, 2000.
- 519 [14] X. Descombes, Multiple objects detection in biological images using a
520 marked point process framework, *Methods* 115 (2017) 2–8. doi:10.1016/
521 j.ymeth.2016.09.009.
- 522 [15] Y. Verdié, F. Lafarge, Detecting parametric objects in large scenes by
523 Monte Carlo sampling, *Int. J. of Comput. Vis.* 106 (1) (2014) 57–75.
524 doi:10.1007/s11263-013-0641-0.
- 525 [16] C. Lacoste, X. Descombes, J. Zerubia, Point processes for unsupervised line
526 network extraction in remote sensing, *IEEE Trans. on Pattern Anal. and*
527 *Mach. Intell.* 27 (10) (2005) 1568–1579. doi:10.1109/TPAMI.2005.206.
- 528 [17] M. Ortner, X. Descombes, J. Zerubia, A Marked Point Process of Rect-
529 angles and Segments for Automatic Analysis of Digital Elevation Models,

- 530 IEEE Trans. on Pattern Anal. and Mach. Intell. 30 (1) (2008) 105–119.
531 doi:10.1109/TPAMI.2007.1159.
- [18] P. Craciun, M. Ortner, J. Zerubia, Joint detection and tracking of moving
532 objects using spatio-temporal marked point processes, in: IEEE Winter
533 Conference on Applications of Computer Vision, 2015, pp. 177–184. doi:
534 10.1109/WACV.2015.31.
535
- [19] M. Kulikova, I. Jermyn, X. Descombes, E. Zhizhina, J. Zerubia, A Marked
536 Point Process Model Including Strong Prior Shape Information Applied
537 to Multiple Object Extraction From Images., Int. J. of Comput. Vis. and
538 Image Process. 1 (2) (2011) 1–12. doi:10.4018/ijcvip.2011040101.
539
- [20] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, F. J. Huang, A tutorial on
540 energy-based learning, Predict. Struct. Data 1 (2006).
541
- [21] Y. Du, I. Mordatch, Implicit Generation and Modeling with Energy Based
542 Models, NeurIPS (2019).
543
- [22] Q. Yu, G. Medioni, Multiple-target tracking by spatiotemporal Monte Carlo
544 Markov Chain data association, IEEE Trans. on Pattern Anal. and Mach.
545 Intell. 31 (12) (2009) 2196–2210. doi:10.1109/TPAMI.2008.253.
546
- [23] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for
547 biomedical image segmentation, in: N. Navab, J. Hornegger, W. M. Wells,
548 A. F. Frangi (Eds.), Medical Image Computing and Computer-Assisted
549 Intervention – MICCAI 2015, Springer International Publishing, Cham,
550 2015, pp. 234–241.
551
- [24] Z. Huang, W. Li, X.-G. Xia, R. Tao, A General Gaussian Heatmap La-
552 bel Assignment for Arbitrary-Oriented Object Detection, IEEE Trans. on
553 Image Process. 31 (2022) 1895–1910. doi:10.1109/TIP.2022.3148874.
554
- [25] E. Shelhamer, J. Long, T. Darrell, Fully Convolutional Networks for Seman-
555 tic Segmentation, IEEE Trans. on Pattern Anal. and Mach. Intell. 39 (4)
556 (2017) 640–651. doi:10.1109/TPAMI.2016.2572683.
557

- 558 [26] D. Neven, B. D. Brabandere, M. Proesmans, L. Van Gool, Instance segmen-
559 tation by jointly optimizing spatial embeddings and clustering bandwidth,
560 in: IEEE/CVF Conference on Computer Vision and Pattern Recognition,
561 2019, pp. 8829–8837. doi:10.1109/CVPR.2019.00904.
- 562 [27] J. Mabon, M. Ortner, J. Zerubia, CNN-Based Energy Learning for MPP
563 Object Detection in Satellite Images, in: IEEE International Workshop
564 on Machine Learning for Signal Processing, 2022, pp. 1–6. doi:10.1109/
565 MLSP55214.2022.9943312.
- 566 [28] G. E. Hinton, Training products of experts by minimizing contrastive
567 divergence, *Neural Comput.* 14 (8) (2002) 1771–1800. doi:10.1162/
568 089976602760128018.
- 569 [29] Y. W. Teh, M. Welling, S. Osindero, G. E. Hinton, Energy-based models
570 for sparse overcomplete representations, *J. of Mach. Learn. Res.* 4 (Dec)
571 (2003) 1235–1260.
- 572 [30] L. Bottou, Stochastic gradient descent tricks, in: G. Montavon, G. B.
573 Orr, K.-R. Müller (Eds.), *Neural Networks: Tricks of the Trade: Second*
574 *Edition*, Springer Berlin Heidelberg, 2012, pp. 421–436. doi:10.1007/
575 978-3-642-35289-8_25.
- 576 [31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan,
577 T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative
578 style, high-performance deep learning library, *Adv. in neural inf. process.*
579 *syst.* 32 (2019).
580 URL <https://pytorch.org/>
- 581 [32] P. J. Green, Reversible jump Markov chain Monte Carlo computation and
582 Bayesian model determination, *Biometrika* 82 (4) (1995) 711–732.
- 583 [33] P. J. Green, D. I. Hastie, Reversible jump MCMC, *Genetics* 155 (3) (2009)
584 1391–1403.

- 585 [34] T. Guilmeau, E. Chouzenoux, V. Elvira, Simulated annealing: a review
586 and a new scheme, in: *IEEE Statistical Signal Processing Workshop (SSP)*,
587 2021, pp. 101–105. doi:10.1109/SSP49050.2021.9513782.
- 588 [35] Y. Verdié, F. Lafarge, Efficient Monte Carlo sampler for detecting para-
589 metric objects in large scenes, in: A. Fitzgibbon, S. Lazebnik, P. Per-
590 ona, Y. Sato, C. Schmid (Eds.), *Computer Vision – ECCV*, Springer
591 Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 539–552. doi:10.1007/
592 978-3-642-33712-3_39.
- 593 [36] M. Miller, U. Grenander, J. OSullivan, D. Snyder, Automatic target recog-
594 nition organized via jump-diffusion algorithms, *IEEE Trans. on Image Proc-
595 ess.* 6 (1) (1997) 157–174. doi:10.1109/83.552104.
- 596 [37] F. Lafarge, G. Gimel’farb, X. Descombes, Geometric feature extraction by
597 a multimarked point process, *IEEE Trans. on Pattern Anal. and Mach.
598 Intell.* 32 (9) (2010) 1597–1609. doi:10.1109/TPAMI.2009.152.
- 599 [38] G.-S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datecu, M. Pelillo,
600 L. Zhang, DOTA: A large-scale dataset for object detection in aerial images,
601 in: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*,
602 2018, pp. 3974–3983. doi:10.1109/CVPR.2018.00418.
- 603 [39] X. Yang, J. Yan, On the Arbitrary-Oriented Object Detection: Classifica-
604 tion Based Approaches Revisited, *Int. J. of Comput. Vis.* 130 (5) (2022)
605 1340–1365. doi:10.1007/s11263-022-01593-w.
- 606 [40] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon,
607 K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Z. Yifu, C. Wong, A. V,
608 D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck,
609 tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, M. Jain,
610 ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation
611 (Nov. 2022). doi:10.5281/zenodo.7347926.