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 14 Mar 2024 (v2)

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 network[1].

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

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

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

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

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

2. Point Process for object detection

2.1. Markov Marked Point Process

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

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

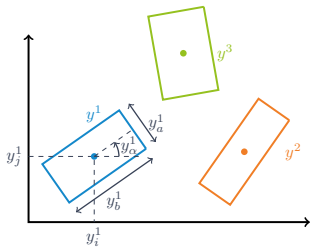


Figure 1: Example configuration with three points; $Y = \{y^1, y^2, y^3\}$.

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

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:

$$h\left(Y\right) \propto \exp\left(-U\left(Y\right)\right), \tag{1}$$

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

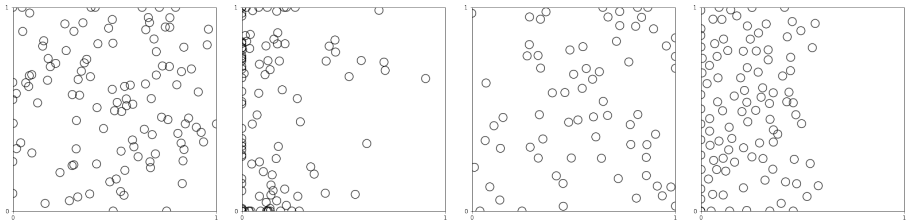
With $V(y|\mathcal{N}_y)$ the energy of point y given the neighborhood of y in Y denoted $\mathcal{N}_y$, the formulation in Eq. 2 ensures the Point Process is Markovian; A Point Process is Markovian under the relationship[2] $\sim$, if and only if $\forall Y \in \mathcal{Y}$ such that $h(Y) > 0$:

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

[2]Note 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 $\mathcal{N}_y$.

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

2.2. Point process for object detection



1) uniform **2)** gradient **3)** repulsive **4)** gradient+repulsive

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

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

The energy formulation of the point process allows compositing several simple point behaviors/distributions into one model. We illustrate the composition of several simple point distributions through the addition of energy terms in Figure 2.

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

sum[16, 17, 18]:

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

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

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

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

- *Prior terms* (of the form $V_k(y|\mathcal{N}_y,\theta)$) that only depend on y and its neighborhood $\mathcal{N}_y$; they measure the coherence of the configuration itself considering the known properties of the objects of interest (e.g., objects don't overlap).

- *Data terms* ($V_k(y|X,\theta)$) are function of y and the image X (also referred as observation in remote sensing); these terms measure the correspondence of the points with the image.

3. Energy model

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

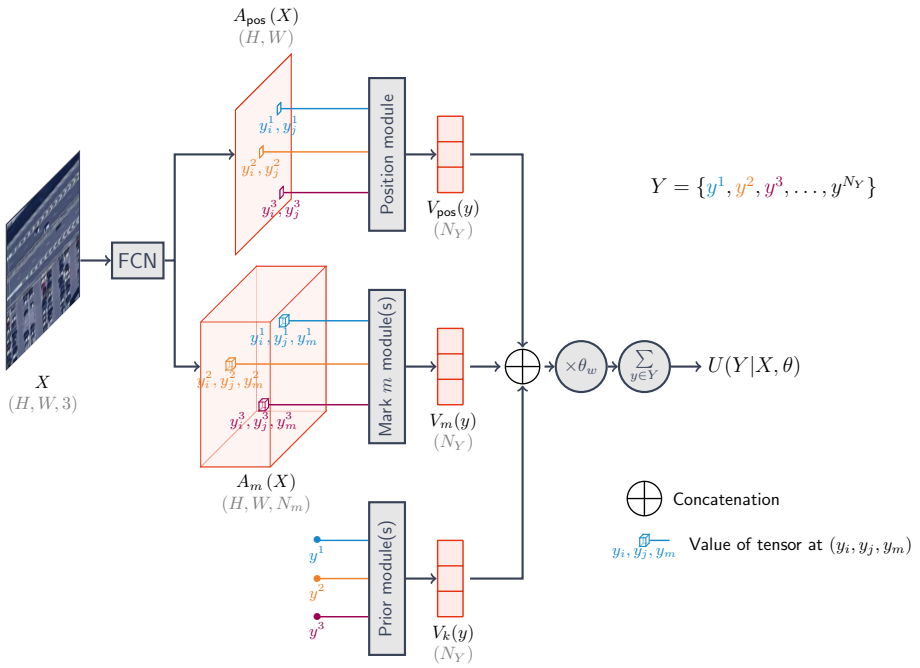


Figure 3: Simplified energy model, with only one mark and prior displayed. With FCN: Fully Convolutional Network, e.g., Unet [23].

3.1. Data terms from CNNs

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

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

3.1.1. Position likelihood term

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

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

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

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

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, $k = 1, \ldots, N_m$, for every discrete value m_k we have:

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

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

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

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

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

3.1.3. FCN for small objects in large images

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

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

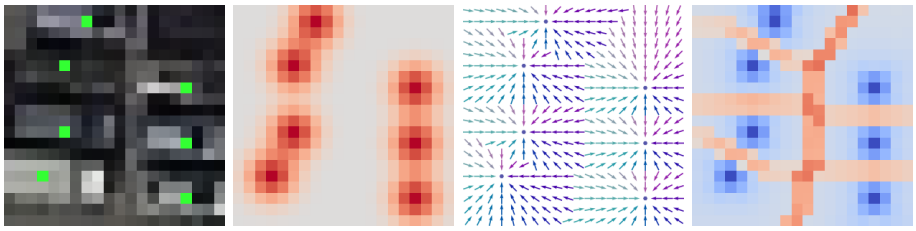
$$A_{\text{pos}}(X) = a.\text{div}\left(A_{\text{vec}}(X)\right) + b. \tag{7}$$

The backbone architecture corresponds to a Unet [23], which outputs for an image X a tensor $F(X)$ with N_F features per pixel. From that feature 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:

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

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



(a) (b) (c) (d)

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

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

3.2. Energy priors

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

3.2.1. Object priors

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

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

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 target value and the standard deviation.

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

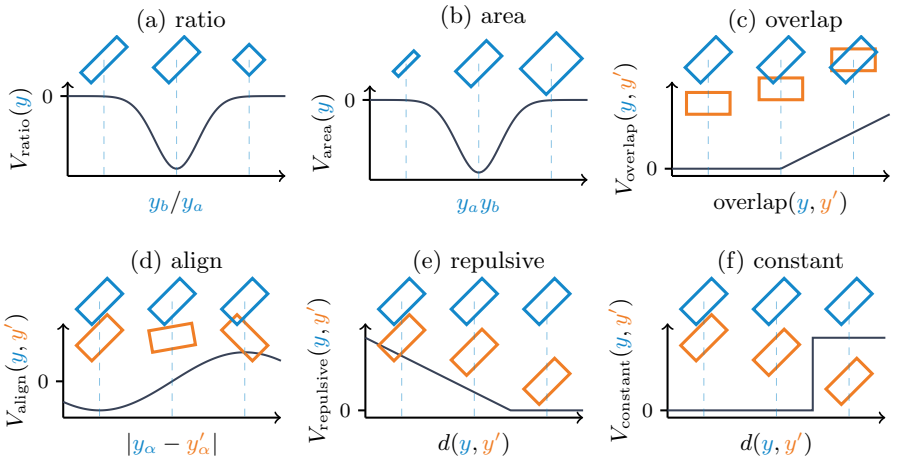


Figure 5: Illustration of some energy priors: (a) and (b) Eq. 10, (c) Eq. 11, (d) Eq. 12, (e) Eq. 13, (f) Eq. 15.

of these priors are illustrated in Figure 5.

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

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

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

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

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

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

3.2.3. Learning prior parameters

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

Eq. 12 can be reformulated as:

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

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

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

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

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

4. Learning model parameters

As introduced in Eq. 3, the energy of a configuration Y will depend on the energy term weights θ_w. The weights θ_w encode the relative importance of the data and prior terms in the point process density. These model parameters can 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). Regardless of the method, the intuition is to select a $\widehat{\theta}$ that approximates θ^* such that:

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

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

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

4.1. Learning energy weights with local perturbations

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

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

$$U\left(Y^+,X,\theta^*\right) < U\left(Y^-,X,\theta^*\right), \quad Y^- \sim Q\left(Y^+ \to \cdot\right) \tag{21}$$

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

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

4.2. Maximum likelihood learning for Energy Based Models

4.2.1. Maximizing likelihood

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

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

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

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

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

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

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

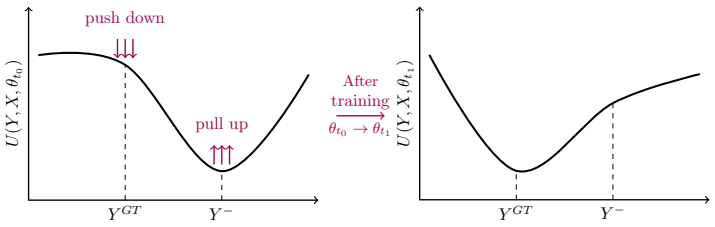


Figure 6: Effect of training the energy with contrastive samples generated from the current θ_n (figure adapted from [20]).

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

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 be approximated through Monte Carlo sampling, where $\widetilde{Y}^0, \ldots, \widetilde{Y}^N$ are drawn from the law defined by $p(\cdot|X_i, \theta_n)$, giving

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

4.2.2. Contrastive divergence

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

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

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

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

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

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

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

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

Algorithm 1 Maximum likelihood learning

$\beta \leftarrow \emptyset$, $n \leftarrow 0$
while not converged **do**
  pick X, Y^{GT} from $\mathcal{S}$
  $\widetilde{Y}_0 \sim \beta_X$ with probability 0.99, else $\mathcal{U}(\mathcal{Y})$
  $Y^- \leftarrow \texttt{SampleMPP}(Y_0, X, \theta_n)$ ▷ Defined in Algorithm 2
  $\beta_X \leftarrow Y^-$
  $\Delta\theta_n \leftarrow \nabla_{\theta_n} \mathcal{L}\left(\theta_n, Y^+, Y^-, X\right)$
  Update θ_{n+1} with $\Delta\theta_n$ using SGD
  $n \leftarrow n + 1$
end while

4.2.3. Relating contrastive divergence to manual method

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

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

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

4.2.4. Learning more than just weights

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

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

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

Meanwhile, the contrastive divergence method (Algorithm 1) requires simulating a few Markov Chain steps in between epochs to generate contrastive samples, 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:

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

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

5. Sampling configurations from the energy model

Given an image X and estimated model parameters $\widehat{\theta}$, $U(Y|X, \widehat{\theta})$ 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 $\widehat{Y}$, that estimate Y^* defined as

$$Y^* = \underset{Y \in \mathcal{Y}}{\operatorname{argmin}} U\left(Y|X, \widehat{\theta}\right). \tag{28}$$

This is achieved by sampling the law derived from $U(Y|X, \widehat{\theta})/T$, with T decreasing geometrically (see Section 5.2).

5.1. Reversible Jump Monte Carlo Markov Chain

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

To speed up convergence Green [32] uses translation/rotation/scaling kernels. Moreover, we add a non-uniform birth kernel that proposes points based on a non-uniform density [16]. This proposal density is referred as a birth map (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]:

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

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

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

5.1.1. Birth and Death kernels

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

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

$$\frac{Q(Y' \to Y)}{Q(Y \to Y')} = \frac{p_{\mathrm{D}} Q_{\mathrm{D}}(Y' \to Y)}{p_{\mathrm{B}} Q_{\mathrm{B}}(Y \to Y')} = \frac{p_{\mathrm{D}}}{p_{\mathrm{B}}} \frac{\lambda |S|}{|Y| + 1} \quad (30)$$

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

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

$$\frac{Q(Y' \to Y)}{Q(Y \to Y')} = \frac{p_{\mathrm{B}} Q_{\mathrm{B}}(Y' \to Y)}{p_{\mathrm{D}} Q_{\mathrm{D}}(Y \to Y')} = \frac{p_{\mathrm{B}}}{p_{\mathrm{D}}} \frac{|Y|}{\lambda |S|} \tag{31}$$

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

5.1.2. Birth maps

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

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

$$\frac{Q(Y' \to Y)}{Q(Y \to Y')} = \frac{p_{\mathrm{D}} Q_{\mathrm{D}}(Y' \to Y)}{p_{\mathrm{B}} Q_{\mathrm{B}}(Y \to Y')} = \frac{\lambda}{d(y)(|Y| + 1)} \tag{32}$$

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 $A_{\mathrm{pos}}(X)$ (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:

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

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

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

Since V_{pos} and V_m are derived from tensors $A_{\mathrm{pos}}(X)$ and $A_m(X)$, we can compute the normalizing constant Z_d over the discrete space of pixel positions[4].

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

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

5.2. Simulated annealing

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

5.3. Parallelization

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

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

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

---

independently.

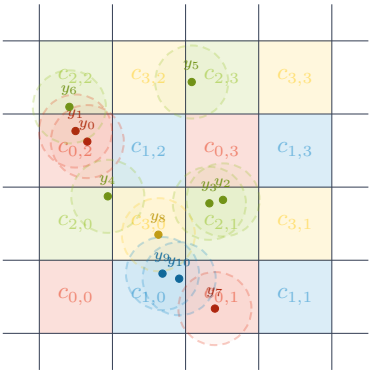


Figure 7: Space partitioning, where each color corresponds to a different mic-set. For cell $c_{s,z}$, is the z^{th} cell of mic-set s. For every point y we display its interaction radius d_{max} as a semi-transparent dotted circle.

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

1. pick a kernel Q_k with probability p_k,
2. pick one mic-set s with probability $p(s)$,
3. keep each cell in mic-set s with probability $p(c|s)$
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

$$Q(Y \to Y') = p_{\text{B}} p(s) p(c|s) Q_{\text{B}}^c(Y \to Y'). \tag{35}$$

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:

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

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

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

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:

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

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

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

The resulting sampling method is outlined in Algorithm 2.

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

for $t = 0, \ldots, N_{\text{smpl}}$ **do**
  Pick *diffusion* with probability 0.8, else *jump*
  Pick one mic-set s with probability $p(s)$
  Keep each c in s with probability $p(c|s)$ to make $\widetilde{s}$
  $Y_{t+1} \leftarrow Y_t$ ▷ unvisited cells keep previous state
  for all $c \in \widetilde{s}$ **do**
    if *diffusion* **then**
      $dw \sim \mathcal{N}(0, \delta)$
      $C_c(Y_{t+1}) \leftarrow C_c(Y_t) - \delta \nabla_{C_c(Y_t)} U(Y_t|X, \theta) + dw\sqrt{2T_t}$
    else
      $Y' \sim Q^c_{BD}(Y_t \to \cdot)$ ▷ perform move in c
      $r \leftarrow \frac{Q(Y_t' \to Y_t)}{Q(Y_t \to Y')} \exp\left(-\frac{\Delta U(Y_t \to Y'|X,\theta)}{T_t}\right)$
      $C_c(Y_{t+1}) \leftarrow C_c(Y')$ with probability $\min(1, r)$
    end if
  end for
  $T_{t+1} \leftarrow \alpha T_t$
end for

6. Application

6.1. Datasets

Our application goal is the detection of small objects in images from satellites such as Pléiades[5] or CO3D[6], which have a typical spatial resolution (or ground sampling distance) of 0.5 meter per pixel.

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

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

[5] Constellation Pléiades [pleiades.cnes.fr]
[6] Constellation Optique 3D [intelligence-airbusds.com]

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

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

6.2. Comparison to PP based on contrast measures

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

To assert the viability of such measures for this data, we compute a score $s(y) = \exp(-V_c(y))$ for proposed detections y (both Ground Truth and randomly scattered rectangles). The true objects represent 1% of this experiment's dataset[7]. A well suited measure $s(y)$ should allow classifying easily between Ground Truth objects and random, non-valid, objects. We plot precision-recall curves for several of those contrast measures in Figure 8 both for a sample image of $\text{DOTA}_{\text{gsd0.5}}$, and for a synthetic highly contrasted image. The Average Prevision (AP) values are reported in Table 1. We compare with the proposed detection energy given by $V_p + V_m$.

The proposed measure, based on CNN output, outperforms classical contrast based measures, as the CNN has learned to differentiate between relevant and irrelevant contrasted objects[8].

[7] A higher proportion of true objects would allow high average precision for useless estimators predicting a constant value.

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

Measure	V_c **formulation**	$\mathbf{AP_{real}}$	$\mathbf{AP_{synthetic}}$
CNN output (ours)	$V_{\text{pos}}(y\|X)\sum_m V_m(y\|X)$	0.99	0.88
T-test [16]	$\frac{\lvert\mu_y-\mu_s\rvert}{\sqrt{\frac{\sigma_y^2}{n_y}+\frac{\sigma_s^2}{n_s}}}$	0.13	0.99
Gradient [19]	$\int n_y(t)\cdot\frac{\nabla X(s(t))}{\sqrt{\lvert\nabla X(s(t))\rvert^2+\epsilon}}dt$	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 ($\text{AP}_{\text{synthetic}}$) from Figure 8

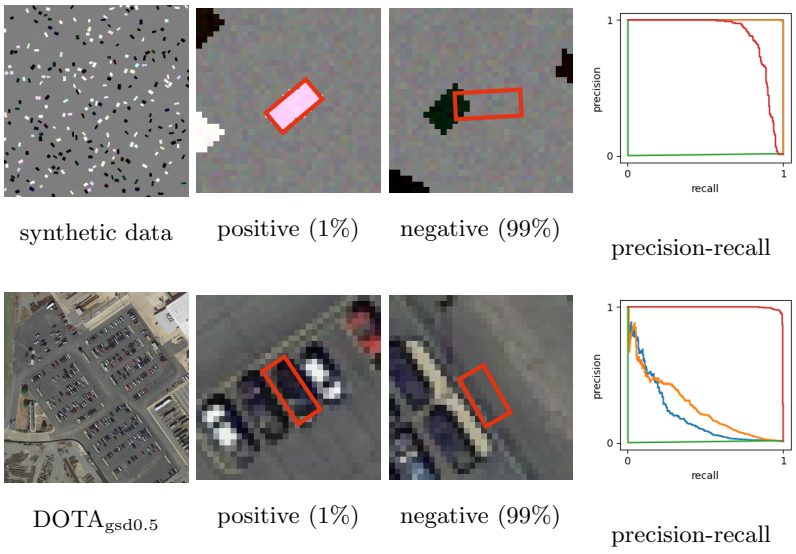


Figure 8: Comparing contrast measures on synthetic (first line) and real data (second line). Positive and negative samples, represent respectively 1% and 99% of the samples for which the metric is computed. For precision-recall; blue: T-test [16], orange: image-gradient[19], red to CNN output, green: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

6.3. Papangelou intensity as confidence score

Classical CNNs for object detection yield a confidence score $s(y) \in \mathbb{R}$ for each proposed object y in the image. This confidence score is often interpreted, for each detection, as proportional to the probability of proposed element y to be a true (labeled) object, $s(y) \propto p(y|X)$. Applying a score threshold t_s gives a set of detections, for which metrics such as precision and recall can be computed. The detector performance over all possible threshold values is summed up via 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 $\widehat{Y}$, $s(y|\widehat{Y} \setminus \{y\}) \propto p(y|\widehat{Y} \setminus \{y\}, X)$. This score is proportional to the Papangelou intensity [13], defined for every $y \in Y$ as:

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

The dependency of the score on the current configuration yields a complication when computing the Average Precision: When applying a threshold t_s to prune the configuration Y into $Y' \subset Y$, for any $y \in Y'$, the score $s(y|Y \setminus \{y\})$ may differ from $s(y|Y' \setminus \{y\})$. With a score of the form $s(y)$, that only depends on y and the image X, such as those from classical CNN models, the score from 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 $Y_1 \supset Y_2 \dots Y_{|\widehat{Y}|-1} \supset Y_{|Y|} \supset \emptyset$, with $Y_1 = \widehat{Y}$:

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

This provides a pruning order $y_1, \dots, y_{|\widehat{Y}|}$ of points in $\widehat{Y}$. This ordering allows to plot the precision and recall curve[9].

[9] Tracing 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.

6.4. Models

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

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

6.5. Results

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

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

Testing the models on noisy data (Figure 9, line 4) and on the ADS data (Figure 10), shows that the added priors allow the PP based models to be 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$^\blacklozenge$	**0.91**	**0.58**
CNN-PP*	**0.92**	**0.62**

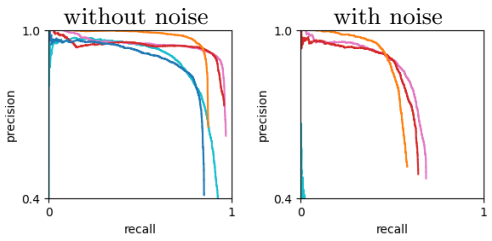


Table 2: Average Precision (AP) values, with AP_0 and AP_N, respectively AP for default and noisy data. Right : Precision Recall (PR) curves, color correspondence in table above.

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

Lastly, CNN-PP$^\blacklozenge$ and CNN-PP* show comparable performances (qualitatively and quantitatively), demonstrating the capability of the contrastive divergence method to infer the prior/interaction model (via the prior parameters), thus limiting the trial and error inputs.

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

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

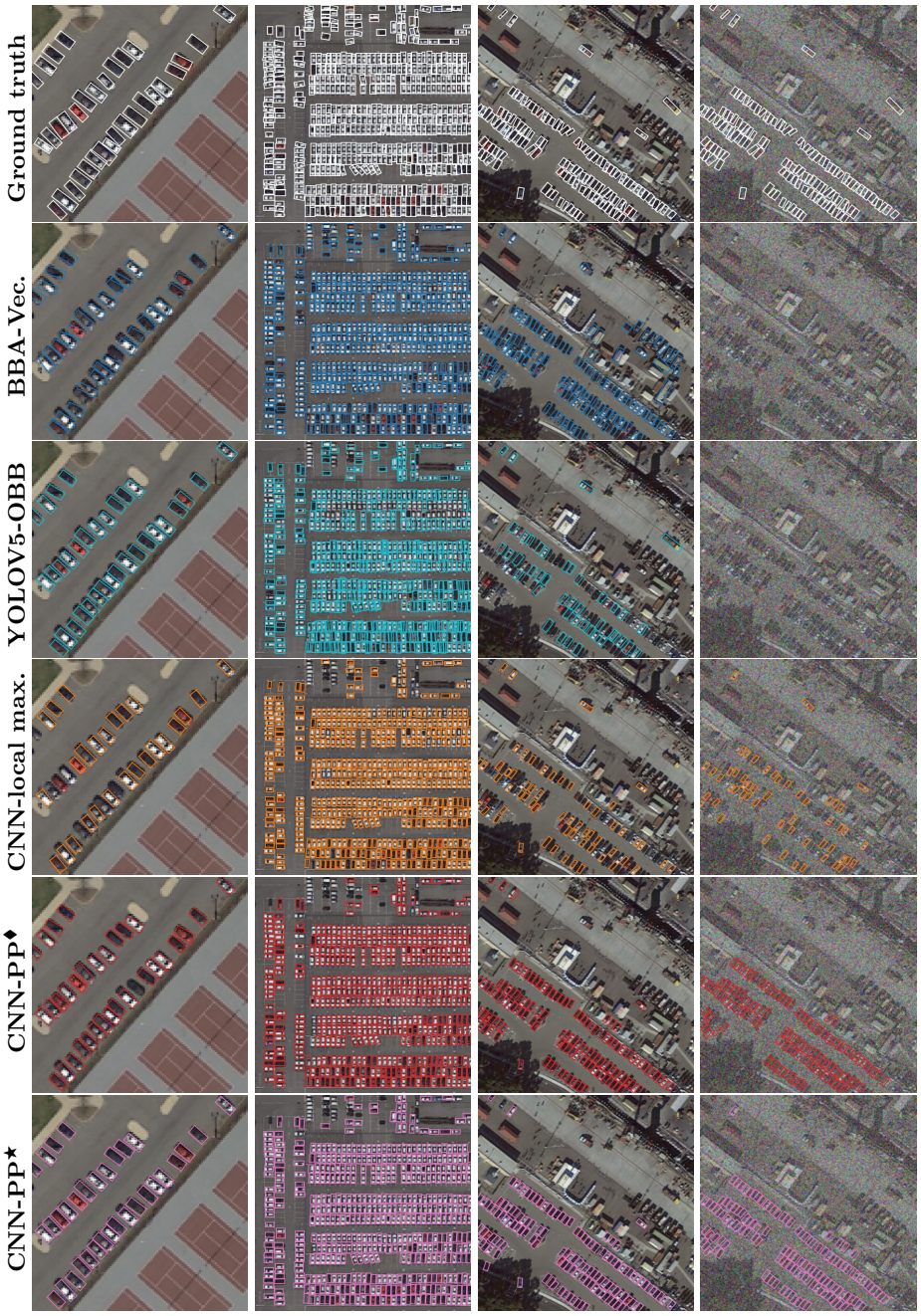


Figure 9: Samples of detection on the test dataset (columns 1-3), and the same dataset with additive Gaussian noise (columns 4). The score threshold (to not display low score objects) is set to maximize the $F1$ score for each model/dataset.

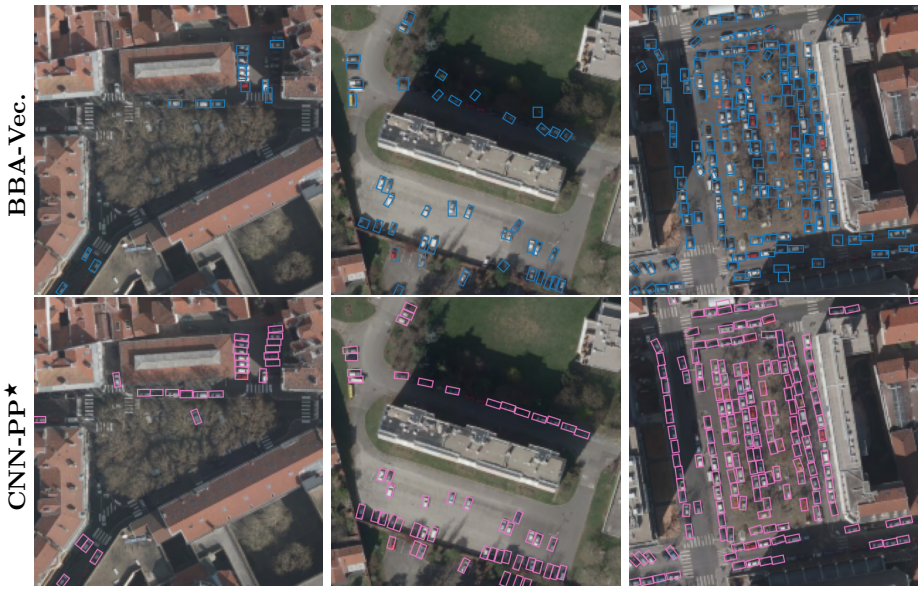


Figure 10: Model applied to ADS data. [© Airbus Defense and Space]

7. Conclusion and perspectives

The proposed method combines Convolutional Neural Networks — that extracts relevant information efficiently from images — with Point Processes — which model interactions with few parameters — allowing to geometrically regularize the inferred configurations. It also improves resilience to input noise compared to mainstream CNN models.

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

With the introduced scoring method based on the Papangelou intensity of the PP, we show our models performs greatly in remote sensing images, especially when the images are noisy.

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

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

References

[1] Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye, Object Detection in 20 Years: A Survey, Proc. IEEE 111 (3) 257–276. doi:10.1109/JPROC.2023.3238524.

[2] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, IEEE Trans. on Pattern Anal. and Mach. Intell. 39 (6) (2017) 1137–1149. doi:10.1109/TPAMI.2016.2577031.

[3] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788. doi:10.1109/CVPR.2016.91.

[4] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, IEEE Trans. on Pattern Anal. and Mach. Intell. 42 (2) (2020) 318–327. doi:10.1109/TPAMI.2018.2858826.

[5] Y. Li, Y. Xing, Z. Wang, T. Xiao, Q. Song, W. Li, J. Wang, A framework of maximum feature exploration oriented remote sensing object detection, IEEE Geosci. and Remote Sens. Lett. 20 (2023) 1–5. doi:10.1109/LGRS.2022.3228689.

[6] Y. Yao, G. Cheng, G. Wang, S. Li, P. Zhou, X. Xie, J. Han, On improving bounding box representations for oriented object detection, IEEE Trans. on Geosci. and Remote Sens. 61 (2023) 1–11. doi:10.1109/TGRS.2022.3231340.

[7] T. Zhao, N. Liu, T. Celik, H.-C. Li, An arbitrary-oriented object detector based on variant Gaussian label in remote sensing images, IEEE Geosci. and Remote Sens. Lett. 19 (2022) 1–5. doi:10.1109/LGRS.2021.3087492.

[8] J. Yi, P. Wu, B. Liu, Q. Huang, H. Qu, D. Metaxas, Oriented object detection in aerial images with Box Boundary-Aware Vectors, in: IEEE Winter Conference on Applications of Computer Vision, 2021, pp. 2149–2158. doi:10.1109/WACV48630.2021.00220.

[9] Y. Yang, X. Tang, Y.-M. Cheung, X. Zhang, F. Liu, J. Ma, L. Jiao, AR2Det: An accurate and real-time rotational one-stage ship detector in remote sensing images, IEEE Trans. on Geosci. and Remote Sens. 60 (2022) 1–14. doi:10.1109/TGRS.2021.3092433.

[10] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444. doi:10.1038/nature14539.

[11] S. He, H. Zou, Y. Wang, R. Li, F. Cheng, ShipSRDet: An end-to-end remote sensing ship detector using super-resolved feature representation, in: IEEE International Geoscience and Remote Sensing Symposium, 2021, pp. 3541–3544. doi:10.1109/IGARSS47720.2021.9554079.

[12] R. LaLonde, D. Zhang, M. Shah, ClusterNet: Detecting small objects in large scenes by exploiting spatio-temporal information, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4003–4012. doi:10.1109/CVPR.2018.00421.

[13] M. Van Lieshout, Markov Point Processes and Their Applications, Imperial College Press, 2000.

[14] X. Descombes, Multiple objects detection in biological images using a marked point process framework, Methods 115 (2017) 2–8. doi:10.1016/j.ymeth.2016.09.009.

[15] Y. Verdié, F. Lafarge, Detecting parametric objects in large scenes by Monte Carlo sampling, Int. J. of Comput. Vis. 106 (1) (2014) 57–75. doi:10.1007/s11263-013-0641-0.

[16] C. Lacoste, X. Descombes, J. Zerubia, Point processes for unsupervised line network extraction in remote sensing, IEEE Trans. on Pattern Anal. and Mach. Intell. 27 (10) (2005) 1568–1579. doi:10.1109/TPAMI.2005.206.

[17] M. Ortner, X. Descombes, J. Zerubia, A Marked Point Process of Rectangles and Segments for Automatic Analysis of Digital Elevation Models,

IEEE Trans. on Pattern Anal. and Mach. Intell. 30 (1) (2008) 105–119. doi:10.1109/TPAMI.2007.1159.

[18] P. Craciun, M. Ortner, J. Zerubia, Joint detection and tracking of moving objects using spatio-temporal marked point processes, in: IEEE Winter Conference on Applications of Computer Vision, 2015, pp. 177–184. doi:10.1109/WACV.2015.31.

[19] M. Kulikova, I. Jermyn, X. Descombes, E. Zhizhina, J. Zerubia, A Marked Point Process Model Including Strong Prior Shape Information Applied to Multiple Object Extraction From Images., Int. J. of Comput. Vis. and Image Process. 1 (2) (2011) 1–12. doi:10.4018/ijcvip.2011040101.

[20] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, F. J. Huang, A tutorial on energy-based learning, Predict. Struct. Data 1 (2006).

[21] Y. Du, I. Mordatch, Implicit Generation and Modeling with Energy Based Models, NeurIPS (2019).

[22] Q. Yu, G. Medioni, Multiple-target tracking by spatiotemporal Monte Carlo Markov Chain data association, IEEE Trans. on Pattern Anal. and Mach. Intell. 31 (12) (2009) 2196–2210. doi:10.1109/TPAMI.2008.253.

[23] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: N. Navab, J. Hornegger, W. M. Wells, A. F. Frangi (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer International Publishing, Cham, 2015, pp. 234–241.

[24] Z. Huang, W. Li, X.-G. Xia, R. Tao, A General Gaussian Heatmap Label Assignment for Arbitrary-Oriented Object Detection, IEEE Trans. on Image Process. 31 (2022) 1895–1910. doi:10.1109/TIP.2022.3148874.

[25] E. Shelhamer, J. Long, T. Darrell, Fully Convolutional Networks for Semantic Segmentation, IEEE Trans. on Pattern Anal. and Mach. Intell. 39 (4) (2017) 640–651. doi:10.1109/TPAMI.2016.2572683.

[26] D. Neven, B. D. Brabandere, M. Proesmans, L. Van Gool, Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8829–8837. doi:10.1109/CVPR.2019.00904.

[27] J. Mabon, M. Ortner, J. Zerubia, CNN-Based Energy Learning for MPP Object Detection in Satellite Images, in: IEEE International Workshop on Machine Learning for Signal Processing, 2022, pp. 1–6. doi:10.1109/MLSP55214.2022.9943312.

[28] G. E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Comput. 14 (8) (2002) 1771–1800. doi:10.1162/089976602760128018.

[29] Y. W. Teh, M. Welling, S. Osindero, G. E. Hinton, Energy-based models for sparse overcomplete representations, J. of Mach. Learn. Res. 4 (Dec) (2003) 1235–1260.

[30] L. Bottou, Stochastic gradient descent tricks, in: G. Montavon, G. B. Orr, K.-R. Müller (Eds.), Neural Networks: Tricks of the Trade: Second Edition, Springer Berlin Heidelberg, 2012, pp. 421–436. doi:10.1007/978-3-642-35289-8_25.

[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, Adv. in neural inf. process. syst. 32 (2019).
URL https://pytorch.org/

[32] P. J. Green, Reversible jump Markov chain Monte Carlo computation and Bayesian model determination, Biometrika 82 (4) (1995) 711–732.

[33] P. J. Green, D. I. Hastie, Reversible jump MCMC, Genetics 155 (3) (2009) 1391–1403.

[34] T. Guilmeau, E. Chouzenoux, V. Elvira, Simulated annealing: a review and a new scheme, in: IEEE Statistical Signal Processing Workshop (SSP), 2021, pp. 101–105. doi:10.1109/SSP49050.2021.9513782.

[35] Y. Verdié, F. Lafarge, Efficient Monte Carlo sampler for detecting parametric objects in large scenes, in: A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, C. Schmid (Eds.), Computer Vision – ECCV, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 539–552. doi:10.1007/978-3-642-33712-3_39.

[36] M. Miller, U. Grenander, J. OSullivan, D. Snyder, Automatic target recognition organized via jump-diffusion algorithms, IEEE Trans. on Image Process. 6 (1) (1997) 157–174. doi:10.1109/83.552104.

[37] F. Lafarge, G. Gimel'farb, X. Descombes, Geometric feature extraction by a multimarked point process, IEEE Trans. on Pattern Anal. and Mach. Intell. 32 (9) (2010) 1597–1609. doi:10.1109/TPAMI.2009.152.

[38] G.-S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datcu, M. Pelillo, L. Zhang, DOTA: A large-scale dataset for object detection in aerial images, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 3974–3983. doi:10.1109/CVPR.2018.00418.

[39] X. Yang, J. Yan, On the Arbitrary-Oriented Object Detection: Classification Based Approaches Revisited, Int. J. of Comput. Vis. 130 (5) (2022) 1340–1365. doi:10.1007/s11263-022-01593-w.

[40] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Z. Yifu, C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, M. Jain, ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation (Nov. 2022). doi:10.5281/zenodo.7347926.