Non-Maximum Suppression for Object Detection by Passing Messages between Windows

Rasmus Rothe[1], Matthieu Guillaumin[1], and Luc Van Gool[1,2]

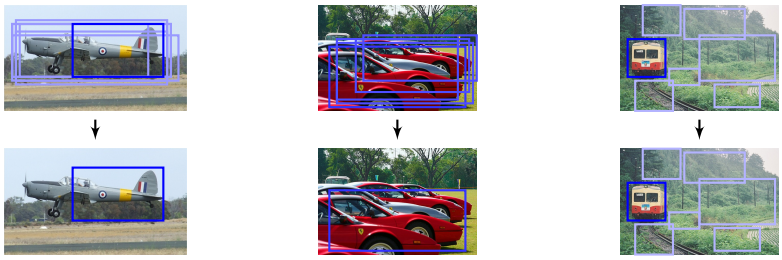
[1] Computer Vision Laboratory, ETH Zurich, Switzerland
{rrothe,guillaumin,vangool}@vision.ee.ethz.ch
[2] ESAT - PSI / IBBT, K.U. Leuven, Belgium
luc.vangool@esat.kuleuven.be

Abstract. Non-maximum suppression (NMS) is a key post-processing step in many computer vision applications. In the context of object detection, it is used to transform a smooth response map that triggers many imprecise object window hypotheses in, ideally, a single bounding-box for each detected object. The most common approach for NMS for object detection is a greedy, locally optimal strategy with several hand-designed components (*e.g.*, thresholds). Such a strategy inherently suffers from several shortcomings, such as the inability to detect nearby objects. In this paper, we try to alleviate these problems and explore a novel formulation of NMS as a well-defined clustering problem. Our method builds on the recent Affinity Propagation Clustering algorithm, which passes messages between data points to identify cluster exemplars. Contrary to the greedy approach, our method is solved globally and its parameters can be automatically learned from training data. In experiments, we show in two contexts – object class and generic object detection – that it provides a promising solution to the shortcomings of the greedy NMS.

1 Introduction

Non-maximum suppression (NMS) has been widely used in several key aspects of computer vision and is an integral part of many proposed approaches in detection, might it be edge, corner or object detection [1–6]. Its necessity stems from the imperfect ability of detection algorithms to localize the concept of interest, resulting in groups of several detections near the real location.

In the context of object detection, approaches based on sliding windows [2–4] typically produce multiple windows with high scores close to the correct location of objects. This is a consequence of the generalization ability of object detectors, the smoothness of the response function and visual correlation of close-by windows. This relatively dense output is generally not satisfying for understanding the content of an image. As a matter of fact, the number of window hypotheses at this step is simply uncorrelated with the real number of objects in the image. The goal of NMS is therefore to retain only one window per group, corresponding to the precise local maximum of the response function, ideally obtaining only one detection per object. Consequently, NMS also has a large positive impact on performance measures that penalize double detections [7, 8].



(a) The top-scoring box may not be the best fit.

(b) It may suppress nearby objects.

(c) It does not suppress false positives.

Fig. 1: Examples of possible failures when using a greedy procedure for NMS. [NB: All our figures are best viewed in color.]

The most common approach for NMS consists of a greedy iterative procedure [2, 3], which we refer to as *Greedy NMS*. The procedure starts by selecting the best scoring window and assuming that it indeed covers an object. Then, the windows that are too close to the selected window are suppressed. Out of the remaining windows, the next top-scoring one is selected, and the procedure is repeated until no more windows remain. This procedure involves defining a measure of similarity between windows and setting a threshold for suppression. These definitions vary substantially from one work to another, but typically they are manually designed. Greedy NMS, although relatively fast, has a number of downsides, as illustrated in Fig. 1. First, by suppressing everything within the neighborhood with a lower confidence, if two or more objects are close to each other, all but one of them will be suppressed. Second, Greedy NMS always keeps the detection with the highest confidence even though in some cases another detection in the surrounding might provide a better fit for the true object. Third, it returns all the bounding-boxes which are not suppressed, even though many could be ignored due to a relatively low confidence or the fact that they are sparse in a subregion within the image.

As these problems are due to greediness and hard-thresholding, in this paper we propose to consider NMS as a clustering problem that is solved globally, where the hard decisions taken by Greedy NMS are replaced with soft penalties in the objective function. The intuition behind our model is that the multiple proposals for the same object should be grouped together and be represented by just one window, the so-called *cluster exemplar*. We therefore adopt the framework of Affinity Propagation Clustering (APC) [9], an exemplar-based clustering algorithm, which is inferred globally by passing messages between data points.

However, APC is not directly usable for NMS. We need to adapt it to include two constraints that are specific to detection. First, since there are false positives, not every window has to be assigned to a cluster. Second, in certain scenarios it is beneficial to encourage a diverse set of proposals and penalize selecting exemplars that are very close to each other. Hence, our contributions are the following: (i) we extend APC to add repellence between cluster centers; (ii) to model false positives, we relax the clustering problem; (iii) we introduce weights

between the terms in APC, and show how these weights can be learned from training data.

We show in our experiments that our approach helps to address the limitations of Greedy NMS in two different contexts: object class detection (Sec. 4) and generic object detection (Sec. 5).

2 Related Work

NMS is a widely used post-processing technique in several computer vision applications. For edge, corner and interest point detection, its role is to find the local maxima of a function defined over a pixel scale-space pyramid, and it is common to simply suppress any pixel which is not the maximum response in its neighborhood [1, 10].

Similarly, for object detection, many approaches have been proposed to prune the set of responses that score above the detection threshold. The Viola-Jones detector [4] partitions those responses in disjoint sets, grouping together responses as soon as they overlap, and propose, for each group with enough windows, a window whose coordinates are the group average. Recently, a more common approach has been to adopt a greedy procedure [2, 3, 11] where the top-scoring window is declared an object, then neighboring windows are removed based on a hand-tuned threshold of a manually-designed similarity (distance between centers when the size ratio is within $0.5 - 2$ in [2, 11]; relative size of the intersection of the windows with respect to the selected object window in [3]). Most current object category detection pipelines [12–14], but also generic object detection ones [7], use such a greedy procedure. As explained in the introduction, a greedy approach with manually-set parameters is not fully satisfactory.

Several alternatives have been considered. A first line of work considers the detector response as a distribution, and formulates the goal of NMS as that of finding the modes of this distribution. For instance, mean-shift for a kernel density estimation [15] and mixtures of scale-sensitive Gaussians [16] have been proposed. Although principled, these approaches still select only local maxima and fail to suppress false positive detections.

A second line of approaches includes iterative procedures to progressively remove extraneous windows. In [17], a re-ranking cascade model is proposed where a standard greedy NMS is used at every step to favor sparse responses. In [18], the authors also adopt an iterative procedure. From a base detector model, a more powerful detector is built using local binary patterns that encode the neighborhood of window scores in the target image. The procedure is iterated several times until saturation of the detector. This is very similar to the idea of contextual boosting [19]. These iterative procedures are rather time-consuming, as they involve re-training object detectors at each iteration.

For the special case of object detection performed through voting, NMS can be done implicitly by preventing a vote to be taken multiple times into account. For instance, with Hough Forests [20–22], patches vote for the location of the object center. The location with maximum response is selected as the object, and the votes within a given radius that contribute to the selected center are removed from the Hough space hence preventing double detections.

The same idea applies to part-based voting for detection [23]. However, these approaches are not generic and do not apply to every object detection framework. In [24, 25], the authors propose to include repulsive pairwise terms into the search for high-scoring windows, so as to avoid performing NMS as a post-processing step. The search is performed using branch-and-bound techniques.

As mentioned earlier, Greedy NMS has the potential shortcoming of suppressing occluding or nearby instances. Several works aim at solving this problem in particular. For the problem of pedestrian detection, [26] proposed to learn detection models for couples of person. Unfortunately, this idea scales very unfavorably with the number of elements in a group, and creates new problems for NMS: what should be done when a double-detection and two single detections are found nearby?

A related field of research generalizes the idea of NMS to the problem of detecting multiple object classes at the same time. This is often referred to as *context rescoring* [3, 27]. Those approaches explicitly model co-occurrence and mutual exclusion of certain object classes, and can incorporate NMS and counts for a given object class [27]. Several works go even further and also model scene type and pixel-level segmentation jointly [28, 29].

To the best of our knowledge, our work is the first to view NMS as a message-passing clustering problem. Clustering algorithms like k-means [30], k-medoids [31] and spectral clustering [32] are not well suited because they return a fixed number of clusters. However, the number of objects and therefore ideal number of clusters is an unknown prior and thus should not have to be fixed in advance. This inflexibility results in poor performance as shown in the experiments. We overcome these limitations by building our approach upon Affinity Propagation Clustering (APC), an exemplar-based clustering approach by Frey [9]. APC has been applied to a variety of problems [33–36] and extended in multiple ways. [37] uses hard cannot-link constraints between two data points which should not be in the same cluster. Our repellence is much weaker and hence more flexible: it penalizes only when two data points are simultaneously cluster centers, resulting in an significantly different formulation than [37].

3 A Message-Passing Approach for NMS

We start in Sec. 3.1 by presenting Affinity Propagation Clustering (APC) [9] using its binary formulation [38], which is the most convenient for our extensions. In Sec. 3.2, we discuss how we have adapted APC for NMS with a novel inter-cluster repellence term and a relaxation of clustering to remove false positives. We show how the messages must be updated to account for these extensions. Finally, in Sec. 3.3, we propose to use a Latent Structured SVM (LSSVM) [39] to learn the weights of APC.

3.1 Affinity Propagation: Binary Formulation and Inference

Let N be the number of data points and $s(i, j)$ the similarity between data points i and $j \in \{1, \ldots, N\}$. APC is a clustering method that relies on data similarities to identify exemplars such that the sum of similarities between exemplars and cluster members is maximized. That is, $s(i, j)$ indicates how well j would serve

as an exemplar for i, usually with $s(i,j) \leq 0$ [9]. Following [38], we use a set of N^2 binary variables c_{ij} to encode the exemplar assignment, with $c_{ij} = 1$ if i is represented by j and 0 otherwise. To obtain a valid clustering, the following constraints must hold: (i) each point belongs to exactly one cluster, or equivalently is represented by a single point: $\forall i : \sum_j c_{ij} = 1$; (ii) when j represents any other point i, then j has to represent itself: $\exists i \neq j : c_{ij} = 1 \Rightarrow c_{jj} = 1$. These constraints can be included directly in the objective function of APC:

$$E_{APC}(\{c_{ij}\}) = \sum_{i,j} S_{ij}(c_{ij}) + \sum_i I_i(c_{i1}, \ldots, c_{iN}) + \sum_j E_j(c_{1j}, \ldots, c_{Nj}), \tag{1}$$

where S_{ij}, I_i and E_j have the following definitions:

$$S_{ij}(c_{ij}) = \begin{cases} s(i,j) & \text{if } c_{ij} = 1 \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

$$I_i(c_{i1}, ..., c_{iN}) = \begin{cases} -\infty & \text{if } \sum_j c_{ij} \neq 1 \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

$$E_j(c_{1j}, ..., c_{Nj}) = \begin{cases} -\infty & \text{if } c_{jj} = 0 \text{ and } \exists i \neq j \text{ s.t. } c_{ij} = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Here I_i enforces (i) while E_j enforces (ii). The *self-similarity* $s(i,i)$ favors certain points to be chosen as an exemplar: the stronger $s(i,i)$, the more contribution it makes to eq. (1).

The inference of eq. (1) is performed by the max-sum message-passing algorithm [9, 38], using two messages: the *availability* α_{ij} (sent from j to i) reflects the accumulated evidence for point i to choose point j as its exemplar, and the *responsibility* ρ_{ij} (sent from i to j) describes how suited j would be as an exemplar for i:

$$\alpha_{ij} = \begin{cases} \sum_{k \neq j} \max(\rho_{kj}, 0) & \text{for } i = j \\ \min(0, \rho_{jj} + \sum_{k \notin \{i,j\}} \max(\rho_{kj}, 0)) & \text{for } i \neq j \end{cases} \tag{5}$$

$$\rho_{ij} = s(i,j) - \max_{q \neq j}(s(i,q) + \alpha_{iq}). \tag{6}$$

3.2 Adapting Affinity Propagation for NMS

We use the windows proposed by the object detector as data points for APC. The self-similarity, or preference to be selected as an exemplar, is naturally chosen as a function of the score of the object detector: the stronger the output, the more likely a data point should be selected. The similarity between two windows is based on their *intersection over union* (IoU), as $s(i,j) = \frac{|i \cap j|}{|i \cup j|} - 1$. Here the indices refer to the area of the windows. This expresses the degree of common area they cover in the image compared to the total area covered which is a good indicator of how likely they describe the same object. To perform competitively, in the following subsections we will extend APC to better suit our needs and present the contributions of this paper. The resulting processing pipeline is depicted in Fig. 2.

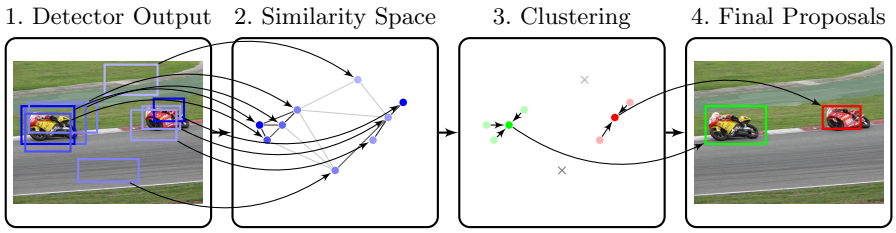


Fig. 2: Illustration of our NMS pipeline. *1. Detector Output*: the detector returns a set of object window hypotheses with scores. *2. Similarity Space*: the windows are mapped into a similarity space expressing how much they overlap. The intensity of the node color denotes how likely a given box is chosen as an exemplar, the edge strength denotes the similarity. *3. Clustering*: APC now selects exemplars to represent window groups, leaving some windows unassigned. *4. Final Proposals*: the algorithm then returns the exemplars as proposals and removes all other hypotheses.

Identifying False Positives. False positives are object hypotheses that belong in fact to the background. Therefore, they should not be assigned to any cluster or chosen as an exemplar. This forces to relax constraint (i). To avoid obtaining only empty clusters, this relaxation must be compensated by a penalty for not assigning a data point to any cluster. We do this by modifying eq. (3):

$$\widetilde{I}_i(c_{i1}, ..., c_{iN}) = \begin{cases} -\infty & \text{if } \sum_j c_{ij} > 1 \\ \lambda & \text{if } \sum_j c_{ij} = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Note how this updated term in eq. (1) is equivalent to adding an extra *background* data point that has similarity λ to all the other data points and 0 self-similarity. In the following, the term $\widetilde{I}_i$ will be weighted, hence we can set $\lambda = -1$ without loss of generality.

Inter-Cluster Repellence. In generic object detection the detector precision is much lower compared to detectors trained for a specific object class. To still achieve a high recall it is beneficial to propose a diverse set of windows that covers a larger fraction of the image. However by default, APC does not explicitly penalize choosing exemplars that are very close to each other, as long as they represent their respective clusters well. To encourage diversity among the windows, we therefore propose to include such a penalty by adding an extra term to eq. (1).

While this term will favor not selecting windows in the same neighborhood, it will not preclude it strictly either. This will still allow APC to select multiple objects in close vicinity. We denote by $R = \sum_{i \neq j} R_{ij}(c_{ii}, c_{jj})$ the new set of *repelling* local functions, where, for $i \neq j$:

$$R_{ij}(c_{ii}, c_{jj}) = \begin{cases} r(i,j) & \text{if } c_{ii} = c_{jj} = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

In other words, we have added a new term for every pair of data points which is active only if both points are exemplars. We penalize this pair by the amount of

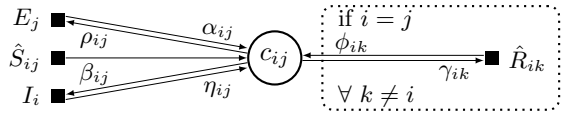


Fig. 3: The 6 messages passed between variables in our extension of Affinity Propagation are α, β, ρ, η, γ and ϕ.

$r(i,j)$, a *repellence cost*. Again, we base the *repellence cost* between two windows on their *intersection over union*, as $r(i,j) = -\frac{|i \cap j|}{|i \cup j|}$. Note that R_{ij} and R_{ji} refer to the same local function. However we keep both notations for simplicity.

Weights and message passing. Linearly combining all the above local functions gives us the following new objective function for APC:

$$\widetilde{E}_{APC} = w_a \sum_i S_{ii} + w_b \sum_{i \neq j} S_{ij} + w_c \sum_i \widetilde{I}_i + w_d \sum_{i<j} R_{ij} + \sum_j E_j. \tag{9}$$

We have omitted the c_{ij} variables for the sake of clarity, and we have further separated data similarities and self-similarities. Note that the local functions are defined so that all weights are expected to be positive.

Weights are only added to the 4 finite terms and only their relative weight matters for inference. Similar to the original APC, we perform inference, *i.e.*, find the values of $\{c_{ij}\}$ that maximize eq. (9) using message-passing. In short, the new terms in eq. (9), especially the repellence ones, lead to new messages to be passed between windows. For the sake of space, we show the factor graph corresponding to eq. (9) and the full derivation of the 6 corresponding messages in the supplementary material. We illustrate them in Fig. 3.

The 6 messages (α, β, ρ, η, γ and ϕ) are reduced to 4 (α, ρ, γ and ϕ) by using substitution and integrating the weights back into the local functions. We view the *background* data point as the N+1-th entry in the similarity matrix and can thereby further simplify the derivation for the message passing. Then we have 2 messages for all variables c_{ij}:

$$\rho_{ij} = \begin{cases} \hat{s}(i,i) - \max\limits_{q \neq i}(\hat{s}(i,q) + \alpha_{iq}) + \sum_{l \neq i} \phi_{il} & \text{for } i = j \\ \hat{s}(i,j) - \max(\max\limits_{q \notin \{i,j\}}(\hat{s}(i,q) + \alpha_{iq}), \hat{s}(i,i) + \alpha_{ii} + \sum_{l \neq i} \phi_{il}) & \text{for } i \neq j, \end{cases} \tag{10}$$

$$\alpha_{ij} = \begin{cases} \sum_{k \neq j} \max(\rho_{kj}, 0) & \text{for } i = j \\ \min(0, \rho_{jj} + \sum_{k \notin \{i,j\}} \max(\rho_{kj}, 0)) & \text{for } i \neq j. \end{cases} \tag{11}$$

Additionally, we have 2 messages essentially resulting from the new R_{ij} term which only exist between the subset $\{c_{ii}\}$ of variables:

$$\gamma_{ik} = \hat{s}(i,i) + \alpha_{ii} - \max_{q \neq i}(\hat{s}(i,q) + \alpha_{iq}) + \sum_{l \notin \{i,k\}} \phi_{il} \tag{12}$$

$$\phi_{ik} = \max(0, \gamma_{ki} + \hat{r}(i,k)) - \max(0, \gamma_{ki}). \tag{13}$$

Following the original message-passing algorithm for APC [9, 38], we initialize all messages with 0. We then iteratively update the messages until convergence.

3.3 Structured Learning for Affinity Propagation

We address now the problem of learning the weights w_a, w_b, w_c and w_d of eq. (9) from training data so as to maximize the performance of the NMS procedure. The training data consists of images with N object window hypotheses and K ground-truth bounding-box annotations for the corresponding object category. The best possible output $\{c^*_{ij}\}$ of APC for those ground-truth bounding-boxes is to keep the proposal with the highest overlap for each ground-truth bounding-box as long as its IoU is at least 0.5. All other proposal should be discarded. This directly determines the target values c^*_{ii} of all c_{ii}. However, correctly setting target values for the remaining c_{ij} $(i \neq j)$ is not straightforward, as we cannot automatically decide which object was detected by this imprecise localization, or whether this window is better modeled as a false positive. Hence, we treat c_{ij} for $i \neq j$ as latent variables. This splits the set of variables in two subsets for each image n: $y_n = \{c^n_{11}, c^n_{22}, ..., c^n_{NN}\}$ are the *observed* variables, with their target y^*_n, and $z_n = \{c^n_{12}, ..., c^n_{1N}, c^n_{21}, c^n_{23}, ..., c^n_{N-1,N}\}$ the *latent* ones.

We can now rewrite our objective function for image n as:
$\widetilde{E}^n_{APC}(y_n, z_n; \boldsymbol{w}) = \boldsymbol{w}^\top \Psi_n(y_n, z_n)$, where Ψ_n is the concatenation of the terms in eq. (9) in a vector, and $\boldsymbol{w} = [w_a, w_b, w_c, w_d, 1]^\top$. To learn $\boldsymbol{w}$, we resort to Structured-output SVM with latent variables (LSSVM) [39]. This consists of the following optimization problem:

$$\begin{aligned} &\operatorname{argmin}_{\boldsymbol{w}\in\mathbb{R}^D, \xi\in\mathbb{R}^n_+} \ \frac{\lambda}{2}||\boldsymbol{w}||^2 + \sum_n \xi^n \\ &\text{s.t. } \forall n, \max_{\hat{z}_n} \widetilde{E}^n_{APC}(y^*_n, \hat{z}_n; \boldsymbol{w}) \geq \max_{y_n, z_n} \left(\widetilde{E}^n_{APC}(y_n, z_n; \boldsymbol{w}) + \Delta(y_n, y^*_n)\right) - \xi^n, \end{aligned} \tag{14}$$

where ξ^n are slack variables, and Δ is a loss measuring how y_n differs from y^*_n. This is equivalent to finding a $\boldsymbol{w}$ which maximizes the energy of APC for the target variables y^*_n, by a margin Δ, independent of the assignment of z_n. Following [39], we solve eq. (14) using the concave-convex procedure (CCCP) [40] and the Structured-output SVM implementation by [41]. We define Δ:

$$\Delta(y, y^*) = \sum_i \nu[c_{ii} - c^*_{ii} < 0] + \pi \left(1 - \max_{\text{obj}} \frac{|i \cap obj|}{|i \cup obj|}\right)[c_{ii} - c^*_{ii} > 0]. \tag{15}$$

where $\nu \geq 0$ is the cost for not choosing a window as an exemplar although it is the best candidate for one of the objects. When a box is chosen as an exemplar even though it is not the best candidate it is considered as a false positive. This is smoothly penalized by $\pi \geq 0$ by considering the overlap with the ground-truth object it most overlaps with. The values for π and ν are chosen depending on the application, usually $\nu/\pi > 1$. Using CCCP additionally implies that we are able to perform loss-augmented inference (*i.e.*, find (y_n, z_n) that maximizes the right-hand side of the constraints in eq. (14)), and partial inference of z_n (*i.e.*, the left-hand side of the constraint). For the left-hand side, $\operatorname{argmax}_{\hat{z}} \widetilde{E}_{APC}(y^*, \hat{z}; \boldsymbol{w})$ can be computed directly. Given the cluster centers y^*_n we just assign all other boxes which are not cluster centers to the most similar clusters. For false positives, this could also be the *background* data point depending on the current value for

w_c. This results in a valid clustering which maximizes the total similarity for the given exemplars.

Concerning the right-hand side, we can easily incorporate Δ as an extra term in eq. (9), and use message passing to obtain the corresponding (y_n, z_n). When incorporating the loss term into the message passing, only the similarity $\hat{s}$ needs to be modified, leading to $\hat{s}_\Delta$:

$$\hat{s}_\Delta(i,j) = \begin{cases} \hat{s}(i,j) - \nu & \text{for } i{=}j \text{ and } c^n_{ii}{=}1 \\ \hat{s}(i,j) + \pi \left(1 - \max\limits_{\text{obj}} \frac{|i \cap obj|}{|i \cup obj|}\right) & \text{for } i{=}j \text{ and } c^n_{ii} = 0 \\ \hat{s}(i,j) & \text{otherwise.} \end{cases} \tag{16}$$

4 Experiments on Object Class Detection

To compare the proposed exemplar based clustering framework to Greedy NMS, we measured their respective performance for object class detection. We are especially interested in the cases we presented in Fig. 1 where Greedy NMS fails, and we will present insights why our proposed method handles these better. A detailed analysis will address localization errors (Fig. 1a), close-by labeled objects (Fig. 1b), precision as well as detections on background (Fig. 1c). This is in line with Hoiem's [42] in-depth analysis of the performance of a detector, not only giving a better understanding of its weaknesses and strengths but also showing that specific improvements are necessary to advance in object detection.

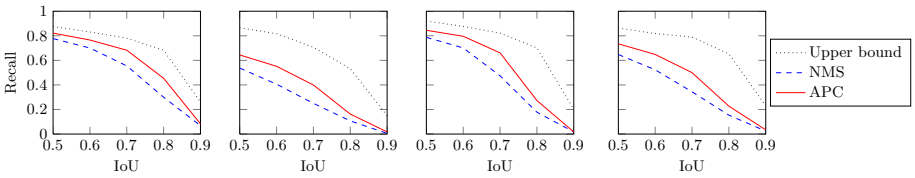
4.1 Implementation Details

In this section the clustering is applied to Felzenszwalb's [3] (release 5) object class detector based on a deformable parts model (DPM). Performance is measured on the widely used Pascal VOC 2007 [8] dataset composed of 9,963 images containing objects of 20 different classes. We keep the split between training and testing data as described in [3]. The DPM training parameters are set to their default values. We keep all windows with a score above a threshold which is determined for each class during training but at most 250 per image. The similarity between two windows is based on their *intersection over union*, as described in Sec. 3. As the score of the Felzenszwalb boxes p is not fixed to a range, it is scaled to $[-1, 0]$ by a sigmoidal function $s(i,i) = \frac{1}{1+e^{-p}} - 1$. The presented results for *APC* are trained following Sec. 3.3 on the validation set. For a fair comparison, the ratio ν/π was set to yield a total number of windows similar to Greedy NMS.

4.2 Results

The results are presented in separate subsections that compare the performance of APC and Greedy NMS with emphasis on the specific issues presented in Fig. 1.

Can APC provide better fitting boxes than Greedy NMS (Fig. 1a)? Here we show that solving NMS globally through clustering can help to select better fitting bounding-boxes compared to Greedy NMS. We look at the detection rate for different *IoU* thresholds with the object for detection. The upper bound is determined by the detection rate of the detector when returning all windows, *i.e.* without any NMS.



(a) bicycle **(b)** chair **(c)** horse **(d)** average

Fig. 4: Object class detection: IoU vs. recall for a selection of classes (a-c) as well as the average across all (d). Our method consistently outperforms Greedy NMS for different IoU thresholds.

Table 1: Object class detection: area under curve (AUC) for IoU vs. recall.

	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	diningtable
Upper bound	0.592	0.716	0.495	0.476	0.482	0.744	0.663	0.718	0.641	0.600	0.788
NMS	0.303	0.494	0.170	0.187	0.288	0.450	0.432	0.335	0.259	0.312	0.391
APC	**0.426**	**0.589**	**0.297**	**0.260**	**0.333**	**0.552**	**0.498**	**0.432**	**0.361**	**0.426**	**0.556**

	dog	horse	motorbike	person	pottedplant	sheep	sofa	train	tvmonitor	**average**
Upper bound	0.685	0.740	0.727	0.620	0.508	0.497	0.855	0.707	0.702	0.648
NMS	0.265	0.439	0.422	0.320	0.170	0.200	0.470	0.394	0.482	0.339
APC	**0.336**	**0.540**	**0.522**	**0.418**	**0.303**	**0.322**	**0.584**	**0.533**	**0.510**	**0.440**

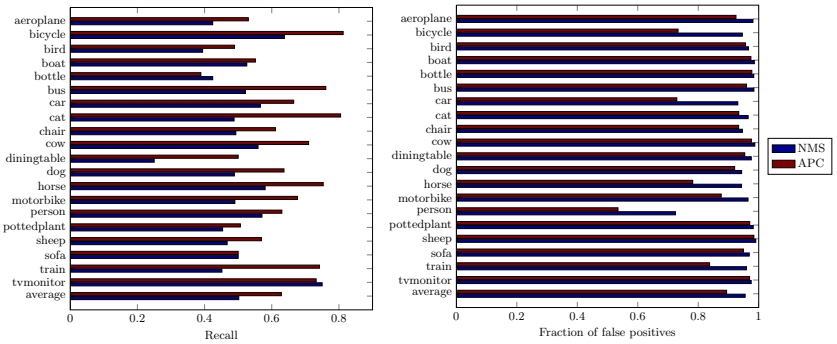
(a) NMS Proposals **(b)** Cluster 1 **(c)** Cluster 2 **(d)** Cluster 3

(e) APC Proposals **(f)** Cluster 1 **(g)** Cluster 2 **(h)** Background

Fig. 5: Object class detection: qualitative results. These figures show an example of the proposed windows. The colored box are the exemplars for the gray boxes. Upper row: Greedy NMS. Lower row: APC.

The quantitative results in Fig. 4 confirm that APC recovers more objects with the same number of boxes compared to Greedy NMS, especially performing well when a more precise location of the object is required ($IoU \geq 0.7$). We then evaluated the area under the curve in Fig. 4 for each class separately (normalized to 1), whose values are shown in Tab. 1. Here we perform better across all classes with an increase between 0.17 for the *diningtable* class and 0.03 for the *tvmonitor* class. On average the AUC can be increased from 0.34 to 0.44. Even though selecting the right boxes from the output of the detector could have led up to an AUC of 0.65, APC was still able to narrow the gap by almost a third.

This is also confirmed by the qualitative results in Fig. 5: whereas NMS proposes several boxes for the same bike (e.g. (b), (c)) and even sometimes proposes one box covering two objects (d), our method returns one box per bike ((f), (g)). These boxes are the exemplars of clusters only containing boxes which tightly fit the bikes – the others are collected in the background cluster (h).



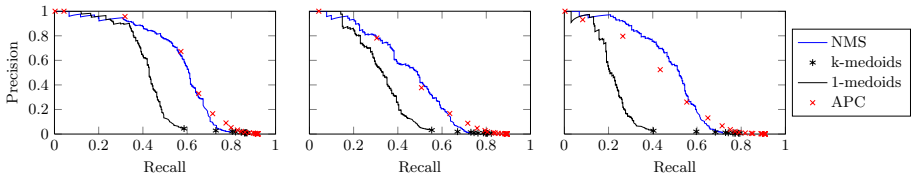
(a) Performance on touching objects **(b)** Suppressing false positives

Fig. 6: Object class detection: in-depth analysis. (a) compares the recall of Greedy NMS and APC on pairs of objects ($IoU > 0$ between objects) – APC recovers significantly more of these rather difficult objects. (b) shows the fraction of false positives – windows that do not touch any object: APC on average reduces the fraction of false positives, with a significant reduction for some classes, *i.e. bicycle, car, person.*

Does APC avoid to suppress objects in groups (Fig. 1b)? Two (or more) objects form a group if they at least touch each other ($IoU > 0$). Thus we remove from the ground-truth the objects that do not overlap with any other object of the same class, and compute the recall (with $IoU = 0.5$) on the remaining objects for the same number of proposed windows as shown in Fig. 6a. On average APC recovers 62.9% objects vs. 50.2% for Greedy NMS, with an increase of up to 31.7% for individual classes. Noting that these objects are especially difficult to detect, APC is more robust at handling nearby detector responses. This is a clear advantage of the proposed clustering based approach.

Can APC suppress more false-positives (Fig. 1c)? Already the qualitative results in Fig. 5h suggest that the clustering relaxation proposed in Sec. 3.2 helps to remove extraneous boxes with low scores which do not describe any object. For a quantitative analysis, we look again at the results of APC and Greedy NMS when both return the same number of windows. Noting that both post-processing algorithms are provided with exactly the same windows by the detector as input, we now evaluate which method is better at suppressing false positives. In this context we define false positives as all boxes which do not touch any object ($IoU = 0$). These boxes are nowhere near detecting an object as usually at least $IoU \geq 0.5$ is required for detection. As shown in Fig. 6b APC is able to reduce the fraction of false positives proposed from 95.5% for NMS to 89.4% with consistent improvement across all classes. For some classes like *bicycle, car* and *person* whose objects often occur next to each other, APC shows significant false positive reduction of up to 21.6%, proposing more relevant windows which also reflects in the recall in Fig. 4.

What is the precision of APC compared to NMS and k-medoids? We now vary the ratio of the training parameters ν/π. APC returns a fixed set of boxes, ranging from less than a box up to several hundreds per image depending on the clustering parameters which are obtained through training



(a) horse **(b)** train **(c)** motorbike

Fig. 7: Object class detection: precision vs. recall. The precision-recall curves reveal that APC performs competitively compared to Greedy NMS at a similar precision but higher recall while significantly outperforming k-medoids.

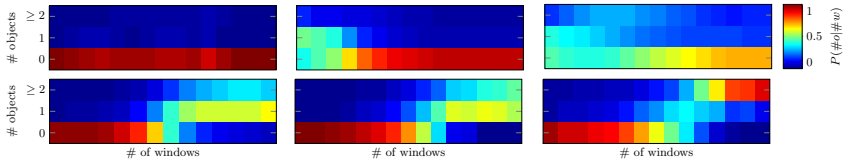
Table 2: Object class detection: average precision NMS vs. APC

		aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	diningtable
IoU 0.5	NMS	**0.332**	**0.593**	0.103	**0.157**	**0.266**	**0.520**	**0.537**	**0.225**	**0.202**	**0.243**	**0.269**
	APC	0.298	0.511	**0.108**	0.107	0.130	0.369	0.428	0.197	0.149	0.168	0.235
IoU 0.8	NMS	**0.101**	0.198	**0.091**	0.023	**0.096**	**0.135**	**0.123**	0.021	0.057	0.048	0.036
	APC	0.090	**0.222**	**0.091**	**0.091**	0.092	0.114	0.112	**0.093**	**0.093**	**0.092**	**0.100**

		dog	horse	motorbike	person	pottedplant	sheep	sofa	train	tvmonitor	**mAP**	**"mAP"**
IoU 0.5	NMS	0.126	0.565	**0.485**	**0.433**	**0.135**	**0.209**	**0.359**	**0.452**	**0.421**	**0.332**	
	APC	**0.129**	**0.579**	0.432	0.363	0.116	0.143	0.259	0.449	0.175		0.267
IoU 0.8	NMS	0.004	0.061	0.126	0.106	0.006	0.030	0.105	0.044	**0.144**	0.078	
	APC	**0.091**	**0.122**	**0.128**	**0.111**	**0.091**	**0.091**	**0.115**	**0.104**	0.107		**0.108**

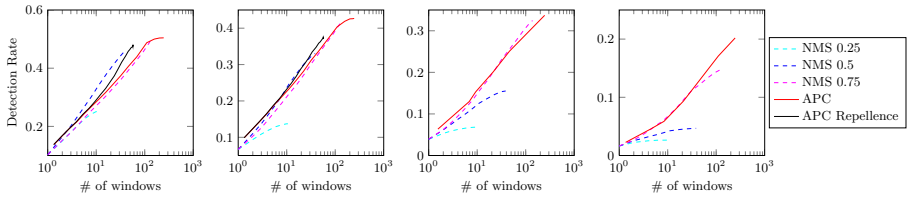
by setting this ratio for the specific application. These boxes, although they cover the objects well, do not follow any kind of ranking as they altogether form the result of a globally solved problem. Since AP is designed to measure the performance of a ranking system, it is simply not appropriate for APC, as that would require that one can select the best possible subset of the proposed boxes. Still, we computed a proxy to AP by linearly interpolating the precision for points of consecutive recall (which need not be consecutive values of the varied parameter). This results in a "mAP" for APC of 0.27 compared to a real mAP of 0.33 for greedy NMS as shown in Tab. 2. AP is mostly influenced by the highest scored detections, so greedy NMS at an IoU of 0.5 is hard to beat with the same underlying detector. However, as such, AP does not reward methods with more precise object localizations than 0.5 and overall better recall. These are precisely areas where greedy NMS can be improved, and therefore we resorted to a deeper analysis. As a matter of fact, if we set a more difficult detection criterion of, e.g., 0.8 IoU, then APC outperforms greedy NMS with a "mAP" of 0.11 compared to 0.08. This is another aspect where APC shows superior performance compared to greedy NMS. As each clustering has a well-defined precision and recall, we can have a scatter plot to compare it to Greedy NMS. Fig. 7 shows that APC achieves a similar precision at low recall but better recall at low precision.

We also compared APC to a k-medoids clustering baseline using the same similarity as for APC. To account for the score of the proposals, the self-similarity of the k selected cluster centers (varied from 1 to 10) was added to the overall cost function to favor boxes with better scores. k-medoids leads to similar precision-recall scatter plots as shown in Fig. 7. Additionally, we plot the precision-recall curve for $k = 1$ (*1-medoids*) by ranking the cluster centers with their original scores. As shown in Fig. 7 already in the case of *1-medoids* many objects are recovered. However, the precision drops for larger recalls since it predicts k objects in every single image. This lack of flexibility is a clear disadvantage of k-medoids and other similar clustering algorithms compared to APC.



(a) bicycle **(b)** car **(c)** person

Fig. 8: Object class detection: predicting the number of objects. Greedy NMS approximately returns the same number of boxes independent of the number of objects in the image. Therefore the posterior $P(\#\,\text{objects}\,|\#\,\text{windows})$ remains uninformative about the object count. In contrast, APC is very flexible and adjust the number of windows being returned depending on how many objects there are in the image.



(a) 0.5 IoU **(b)** 0.6 IoU **(c)** 0.7 IoU **(d)** 0.8 IoU

Fig. 9: Generic object detection: Greedy NMS requires to adopt the parameter for suppression for different IoU thresholds to always perform competitively. In contrast, APC performs consistently well, beating Greedy NMS especially for precise object detection ($IoU \geq 0.7$). Introducing a repellence helps to boost performance for less precise object detection by enforcing diversity among the proposed windows.

Does APC better predict the number of objects in the image? Studying the experimental results revealed that Greedy NMS approximately returns the same number of boxes per image independent of whether there was an object in the image. In contrast, for APC it greatly varied between images. Therefore, we simply measured the posterior probability $P(\#\text{ objects }|\#\text{ windows})$. Fig. 8 depicts this probability for both Greedy NMS and APC for a selection of classes. For Greedy NMS (upper row in Fig. 8) the number of proposed windows is mostly uninformative regarding how many objects there are in the image. In comparison for APC (lower row in Fig. 8), there is a strong correlation between the number of windows proposed and the likelihood that there are 1 or more objects: given the number of windows APC proposes we can estimate how many objects there are in the image.

5 Experiments on Generic Object Detection

We apply APC to generic object detection which gained popularity in recent years as a preprocessing step for many state-of-the-art object detectors [13, 14]. We use the objectness measure introduced by [43] which is the only one to provide a probability p with the window it proposes, unlike [14, 44, 45].

5.1 Implementation Details

Performance is again evaluated on Pascal VOC 2007 where we split the dataset in the same way as in [7] and used the classes *bird, car, cat, cow, dog, sheep*

for training the objectness algorithm as well as the clustering and the remaining 14 classes for testing. Images which had occurrences of both training and testing classes were dropped and in contrast to [7] we also kept objects marked as difficult and truncated. The self-similarity is based on the probability of containing an object $s(i,i) = p(i) - 1$ and the similarity between boxes is defined by the overlap. We sampled 250 windows with multinomial sampling which still allows to recover a large fraction of the objects. As presented in [7], Greedy NMS significantly improved the detection rate for objectness. This motivates our experiments where we compare Greedy NMS against APC.

5.2 Results

After training APC, we compare its detection rate with Greedy NMS for different IoU thresholds with the object. For APC we show the performance both without and with repellence; for NMS we varied the threshold for suppression. Looking at Fig. 9, we make 3 observations: (i) when proposing very few windows per image (< 10) APC typically performs better than Greedy NMS. (ii) for an $IoU \geq 0.7$ the standard NMS threshold of 0.5 performs significantly worse than APC. This requires that Greedy NMS re-runs with a higher threshold for suppression. In comparison our method is much more consistent across varying IoU. (iii) for APC diversity can be enforced by activating the inter-cluster repellence which avoids having cluster centers close-by each other. This boosts our performance for $IoU \leq 0.6$ by close to up to 5% from 42.9% to 47.5% for $IoU = 0.5$.

6 Discussion

We presented a novel clustering-based NMS algorithm based on Affinity Propagation. We showed that it successfully tackles shortcomings of Greedy NMS for object class and generic object detection.

Specifically we show that our method – whose parameters can be learned automatically depending on the application – yields better fitting bounding-boxes, reduces false positives, handles close-by objects better and is better able to predict the number of objects in an image, all at a competitive precision compared to Greedy NMS. Given that APC tries to find a global solution to the NMS problem it is however computationally more complex and still relatively slow taking approximately $1s$ to cluster 250 bounding-boxes. In the future, we therefore plan to explore approximative solutions.

APC could also be expanded to multi-class object detection integrating context and holistic knowledge. The newly introduced repellence could be based not only on the overlap between the boxes but rather the similarity in appearance expressing how likely the two windows cover the same object. In future work, we want to learn the window similarity potentially including visual features that may help to distinguish between multiple detections of the same object or nearby objects. We are convinced that APC can be of interest for many other areas where NMS is used, *e.g.* edge detection [1, 46].

Acknowledgement. The authors gratefully acknowledge support by Toyota.

References

1. Canny, J.: A computational approach to edge detection. TPAMI **8** (6) (1986) 679–698
2. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. CVPR. (2005)
3. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. TPAMI **32** (9) (2010) 1627–1645
4. Viola, P., Jones, M.: Robust real-time object detection. IJCV **57** (2) (2004) 137–154.
5. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. CVPR. (2014)
6. Cheng, M.M., Zhang, Z., Lin, W.Y., Torr, P.H.S.: BING: Binarized normed gradients for objectness estimation at 300fps. CVPR. (2014)
7. Alexe, B., Deselaers, T., Ferrari, V.: Measuring the objectness of image windows. TPAMI **34** (11) (2012) 2189–2202
8. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. IJCV **88** (2) (2010) 303–338
9. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315** (5814) (2007) 972–976
10. Mikolajczyk, K., Schmid, C.: Scale & Affine invariant interest point detectors. IJCV **1** (60) (2004) 63–86
11. Schneiderman, H., Kanade, T.: Object detection using the statistics of parts. IJCV **56** (3) (2004) 151–177
12. Cinbis, R.G., Verbeek, J., Schmid, C.: Segmentation Driven Object Detection with Fisher Vectors. ICCV. (2013)
13. Szegedy, C., Toshev, A., Erhan, D.: Deep neural networks for object detection. NIPS. (2013)
14. Uijlings, J.R.R., van de Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective search for object recognition. IJCV **104** (2) (2013) 154–171
15. Dalal, N.: Finding people in images and videos. PhD thesis, Institut National Polytechnique de Grenoble. (2006)
16. Wojcikiewicz, W.: Probabilistic modelling of multiple observations in face detection. Technical report, Humboldt-Universität zu Berlin (2008)
17. Blaschko, M.B., Kannala, J., Rahtu, E.: Non Maximal Suppression in Cascaded Ranking Models. Scandinavian Conference on Image Analysis (SCIA). (2013)
18. Chen, G., Ding, Y., Xiao, J., Han, T.X.: Detection evolution with multi-order contextual co-occurrence. CVPR. (2013)
19. Ding, Y., Xiao, J.: Contextual boost for pedestrian detection. CVPR. (2012)
20. Razavi, N., Gall, J., Van Gool, L.: Backprojection revisited: Scalable multi-view object detection and similarity metrics for detections. ECCV. (2010)
21. Barinova, O., Lempitsky, V., Kholi, P.: On detection of multiple object instances using hough transforms. TPAMI **34** (9) (2012) 1773–1784
22. Wohlhart, P., Donoser, M., Roth, P. M., Bischof, H.: Detecting partially occluded objects with an implicit shape model random field. ACCV. (2012)
23. Wu, B., Nevatia, R.: Detection and segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses. IJCV **82** (2) (2009) 185–204
24. Blaschko, M.B., Lampert, C.H.: Learning to localize objects with structured output regression. ECCV. (2008)

25. Blaschko, M.B.: Branch and Bound Strategies for Non-maximal Suppression in Object Detection. EMMCVPR. (2013)
26. Tang, S., Andriluka, M., Schiele, B.: Detection and tracking of occluded people. BMVC. (2012)
27. Desai, C., Ramanan, D., Fowlkes, C.C.: Discriminative models for multi-class object layout. IJCV **95** (1) (2011) 1–12
28. Ladicky, L., Sturgess, P., Alahari, K., Russell, C., Torr, P.H.: What, where and how many? combining object detectors and crfs. ECCV. (2010)
29. Yao, J., Fidler, S., Urtasun, R.: Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. CVPR. (2012)
30. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. **1** (14) (1967) 281–297
31. Kaufman, L., Rousseeuw, P.: Clustering by means of medoids. Statistical Data Analysis Based on the L1-Norm and Related Methods. (1987)
32. Von Luxburg, U.: A tutorial on spectral clustering. Statistics and computing. **17** (4) (2007) 395–416
33. Dueck, D., Frey, B.J.: Non-metric affinity propagation for unsupervised image categorization. ICCV. (2007)
34. Dueck, D., Frey, B.J., Jojic, N., Jojic, V., Giaever, G., Emili, A., Musso, G., Hegele, R.: Using Affinity Propagation. RECOMB. (2008)
35. Lazic, N., Frey, B.J., Aarabi, P.: Solving the Uncapacitated Facility Location Problem Using Message Passing Algorithms. AISTATS. (2010)
36. Givoni, I.E., Chung, C., Frey, B.J.: Hierarchical Affinity Propagation. The 27th Conference on Uncertainty in Artificial Intelligence (UAI). (2011)
37. Givoni, I.E., Frey, B.J.: Semi-Supervised Affinity Propagation with Instance-Level Constraints. AISTATS. (2009)
38. Givoni, I.E., Frey, B.J.: A Binary Variable Model for Affinity Propagation. Neural Computation **21** (6) (2009) 1589–1600
39. Yu, C.N.J., Joachims, T.: Learning structural svms with latent variables. ICML. (2009)
40. Yuille, A.L., Rangarajan, A.: The concave-convex procedure. Neural Computation **15** (4) (2003) 915–936
41. Vedaldi, A.: A MATLAB wrapper of SVMstruct. (2011)
42. Hoiem, D., Chodpathumwan, Y., Dai, Q.: Diagnosing Error in Object Detectors. ECCV. (2012)
43. Alexe, B., Deselaers, T., Ferrari, V.: What is an object? CVPR. (2010)
44. Manén, S., Guillaumin, M., Van Gool, L.: Prime Object Proposals with Randomized Prim's Algorithm. ICCV. (2013)
45. Ristin, M., Gall, J., Van Gool, L.: Local context priors for object proposal generation. ACCV. (2012)
46. Dollar, P., Zitnick, C.L.: Structured Forests for Fast Edge Detection. ICCV. (2013)