What is an object ?

Bogdan Alexe, Thomas Deselaers, Vittorio Ferrari
Computer Vision Laboratory, ETH Zurich
{bogdan, deselaers, ferrari}@vision.ee.ethz.ch

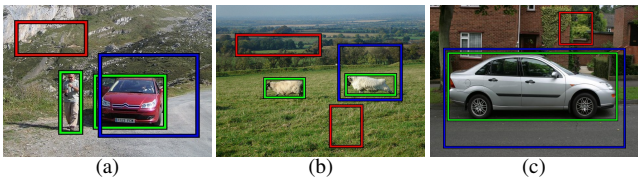
Abstract

We present a generic objectness measure, quantifying how likely it is for an image window to contain an object of any class. We explicitly train it to distinguish objects with a well-defined boundary in space, such as cows and telephones, from amorphous background elements, such as grass and road. The measure combines in a Bayesian framework several image cues measuring characteristics of objects, such as appearing different from their surroundings and having a closed boundary. This includes an innovative cue measuring the closed boundary characteristic. In experiments on the challenging PASCAL VOC 07 dataset, we show this new cue to outperform a state-of-the-art saliency measure [17], and the combined measure to perform better than any cue alone. Finally, we show how to sample windows from an image according to their objectness distribution and give an algorithm to employ them as location priors for modern class-specific object detectors. In experiments on PASCAL VOC 07 we show this greatly reduces the number of windows evaluated by class-specific object detectors.

1. Introduction

In recent years object class detection has become a major research area. Although a variety of approaches exist [1, 4, 22], most state-of-the-art detectors follow the sliding-window paradigm [4, 5, 10, 15, 21]. A classifier is first trained to distinguish windows containing instances of a given class from all other windows. The classifier is then used to score every window in a test image. Local maxima of the score localize instances of the class.

While object detectors are specialized for one object class, such as cars or swans, in this paper we define and train a measure of objectness *generic over classes*. It quantifies how likely it is for an image window to cover an object of *any* class. Objects are standalone things with a well-defined boundary and center, such as cows, cars, and telephones, as opposed to amorphous background stuff, such as sky, grass, and road (as in the 'things *vs*. stuff' distinction of [16]). Fig. 1 illustrates the desired behavior of an objectness measure. It should score highest windows fitting an object tight (green), score lower windows covering partly an object and partly the background (blue), and score lowest windows containing only stuff (red).



(a) (b) (c)

Fig. 1: **Desired behavior of an objectness measure.** *The desired objectness measure should score the blue windows, partially covering the objects, lower than the ground truth windows (green), and score even lower the red windows containing only stuff or small parts of objects.*

We argue that any object has at least one of three distinctive characteristics: (a) a well-defined *closed boundary* in space; (b) a *different appearance* from their surroundings [23, 25]; (c) sometimes it is *unique* within the image and stands out as salient [2, 13, 17, 19]. Many objects have several of these characteristics at the same time (fig. 2-4).

This paper makes three contributions: (a) We design an objectness measure and explicitly train it to distinguish windows containing an object from background windows. This measure combines in a Bayesian framework several image cues based on the above characteristics. (b) We present a new cue (sec. 2.4) and demonstrate it outperforms traditional saliency [17] for detecting objects in the challenging PASCAL VOC 07 dataset [7]. We also show that the combined objectness measure performs better than any cue alone. (c) We show how to use objectness as a *location prior* for modern class-specific object detectors [4, 10, 32]. We demonstrate an algorithm to greatly reduce the number of evaluated windows with only minor loss in detection performance. Different from ESS [21], our method imposes no restriction on the class model used to score a window.

In addition to speeding up detectors, the objectness measure can act as a focus of attention mechanism in other applications. It can facilitate learning new classes in a weakly supervised scenario [6], where the location of object instances is unknown [12, 34]. Similarly, it can help object tracking in video, e.g. incorporated as a likelihood term in a CONDENSATION framework [18].

The source code for the objectness measure is available from http://www.vision.ee.ethz.ch/~calvin.

1.1. Related work

This paper is related to several research strands, which differ in how they define 'saliency'.

Interest points. Interest point detectors (IPs) [20, 27] respond to local textured image neighborhoods and are widely used for finding image correspondences [27] and recognizing specific objects [24]. IPs focus on individual points, while our approach is trained to respond to entire objects. Moreover, IPs are designed for repeatable detection in spite of changing imaging conditions, while our objectness measure is trained to distinguish objects from backgrounds. In sec. 5, we experimentally evaluate IPs on our task.

Class-specific saliency. A few works [26, 28, 33] define as salient the visual characteristics that best distinguish a particular object class (e.g. cars) from others. This class-specific saliency is very different from the class-generic task we tackle here.

Generic saliency. Since [19], numerous works [2, 13, 14, 17] appeared to measure the saliency of pixels, as the degree of uniqueness of their neighborhood w.r.t. the entire image or the surrounding area [23, 25]. Salient pixels form blobs that 'stand out' from the image.

Liu et al. [23] find a single dominant salient object in an image. It combines pixel-based saliency measurements in a CRF and derives a binary segmentation separating the object from the background. Analogously, [31] finds the region with the highest sum of pixel-saliency. These approaches do not seem suitable for the PASCAL VOC dataset [7] where many objects are present in an image and they are not always dominant (fig. 10, 11).

This paper is most related to the above works, as we are looking for generic objects. We incorporate a state-of-the-art saliency detector [17] as one cue into our objectness measure. However, we also include other cues than 'stand out' saliency and demonstrate that our combined measure performs better at finding objects (sec. 5) .

Our work differs from the above also in other respects. The unit of analysis is not a pixel, as possibly belonging to an object, but a window, as possibly containing an entire object. This enables scoring all windows in an image and sampling any desired number of windows according to their scores. These can then directly be fed as useful location priors to object class learning and detection algorithms, rather than making hard decisions early on. We experimentally demonstrate this with an application to speed up object detection (sec. 6).

Analyzing windows also enables evaluating more complex measures of objectness. In sec. 2.4, we propose a new image cue, and demonstrate it performs better than traditional saliency cues [17] at finding entire objects.

Finally, to the best of our knowledge, we are the first to evaluate on a dataset as varied and challenging as PASCAL VOC 07, where most images contain many objects and they appear over a wide range of scales. We explicitly train our objectness measure to satisfy the strict PASCAL-overlap criterion, and evaluate its performance using it. This matters because it is the standard criterion for evaluating the intended clients of our objectness measure, i.e. object detection algorithms.

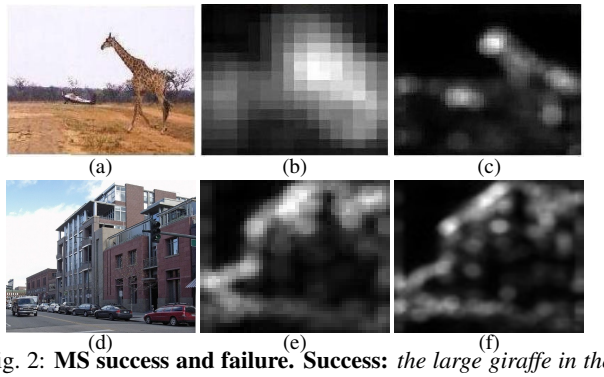


Fig. 2: **MS success and failure. Success:** *the large giraffe in the original image (a) appears as a blob in the saliency map for a high scale (b), while the tiny airplane in the map for a low scale (c). Having multi-scale saliency maps is important for finding more objects in challenging datasets. Interestingly, at the low scale the head of the giraffe is salient, rather than the whole giraffe.* **Failure:** *the numerous cars in the original image (d) are not salient at any scale. We show the saliency maps for 2 scales in (e) and (f). The contour of the building appears more salient than the cars.*

1.2. Plan of the paper.

Sec. 2 describes the image cues composing our objectness measure. Sec. 3 and 4 show how to learn the cue parameters and how to combine them in a Bayesian framework. We present extensive experiments in sec. 5 and show applications of the objectness measure to aid class-specific object detectors [4, 10, 32] in sec. 6.

2. Image cues

As mentioned in sec. 1, objects in an image are characterized by a closed boundary in 3D space, a different appearance from their immediate surrounding and sometimes by uniqueness. These characteristics suggested the four image cues we use in the objectness measure: multi-scale saliency, color contrast, edge density and straddleness.

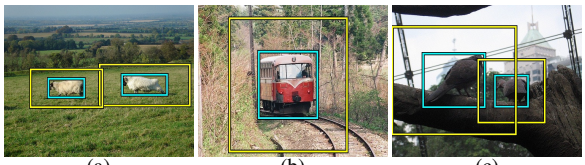
2.1. Multi-scale Saliency (MS).

Hou et al. [17] proposed a global saliency measure based on the spectral residual of the FFT, which favors regions with an unique appearance within the entire image. As it prefers objects at a certain scale, we extend it to multiple scales (fig. 2). Moreover, as [17] suggests, we process the color channels independently as separate images.

For each scale s, we use [17] to obtain a saliency map $I^s_{\mathrm{MS}}(p)$ defining the saliency for every pixel p. Based on this, we define the saliency of a window w at scale s as:

$$\mathrm{MS}(w, \theta^s_{\mathrm{MS}}) = \sum_{\{p \in w | I^s_{\mathrm{MS}}(p) \geq \theta_s\}} I^s_{\mathrm{MS}}(p) \times \frac{|\{p \in w | I^s_{\mathrm{MS}}(p) \geq \theta_s\}|}{|w|} \quad (1)$$

where the scale-specific thresholds θ^s_{MS} are parameters to be learned (sec. 3). Saliency is higher for windows with higher density of salient pixels (second factor), with a bias towards larger windows (first factor). Density alone would score highest windows comprising just a few very salient pixels. Instead, our measure is designed to score highest windows around entire *blobs* of salient pixels, which correspond better to whole objects (fig. 2). The need for multiple



(a) (b) (c)

Fig. 3: **CC success and failure. Success:** *the windows containing the objects (cyan) have high color contrast with their surrounding ring (yellow) in images (a) and (b).* **Failure:** *the color contrast for windows in cyan in image (c) is much lower.*

scales is evident in fig. 2 as the windows covering the two objects in the image (airplane, giraffe) score highest at different scales. This MS cue measures the *uniqueness* characteristic of objects.

2.2. Color Contrast (CC).

The CC cue is a local measure of the dissimilarity of a window to its immediate surrounding area. The surrounding $\text{Surr}(w, \theta_{CC})$ of a window w is a rectangular ring obtained by enlarging the window by a factor θ_{CC} in all directions, so that $\frac{|\text{Surr}(w,\theta_{CC})|}{|w|} = \theta_{CC}^2 - 1$ (fig. 3a-f). The CC between a window and its surrounding is computed as the Chi-square distance between their LAB histograms h:

$$\text{CC}\left(w, \theta_{CC}\right) = \chi^2(h(w),\ h(\text{Surr}(w, \theta_{CC}))) \tag{2}$$

CC is a useful cue because objects tend to have a different appearance (color distribution) than the background behind them. In fig. 3a, windows on the grass score lower than windows half on a sheep and half on the grass. Windows fitting a sheep tightly score highest. This cue measures the *different appearance* characteristic of objects.

CC is related to the center-surround histogram cue of [23]. However, [23] computes a center-surround histogram centered at a pixel, whereas CC scores a whole window as whether it contains an entire object. The latter seems a more appropriate level of analysis.

2.3. Edge Density (ED).

The ED cue measures the density of edges near the window borders. The inner ring $\text{Inn}(w, \theta_{ED})$ of a window w is obtained by shrinking it by a factor θ_{ED} in all directions, so that $\frac{|\text{Inn}(w,\theta_{ED})|}{|w|} = 1/\theta_{ED}^2$. The ED is computed as the density of edgels in the inner ring:

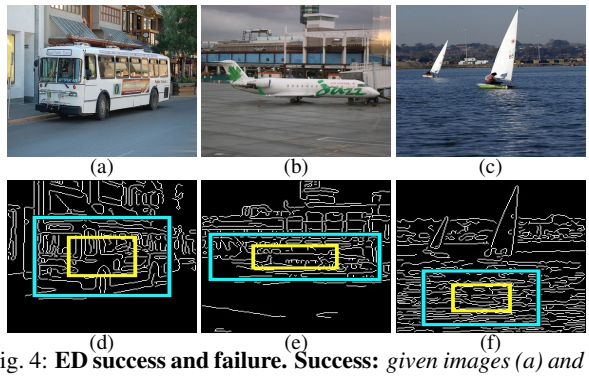
$$\text{ED}(w, \theta_{ED}) = \frac{\sum_{p \in \text{Inn}(w,\theta_{ED})} I_{ED}(p)}{\text{Len}(\text{Inn}(w, \theta_{ED}))} \tag{3}$$

The binary edgemap $I_{ED}(p) \in \{0, 1\}$ is obtained using the Canny detector, and $\text{Len}(\cdot)$ measures the perimeter [1]. The ED cue captures the *closed boundary* characteristic of objects, as they tend to have many edgels in the inner ring (fig. 4d-e).

2.4. Superpixels Straddling (SS).

A different way to capture the *closed boundary* characteristic of objects rests on using superpixels [11] as features.

[1]The expected number of boundary edgels grows proportionally to the perimeter, not the area, because edgels have constant thickness of 1 pixel.



(a) (b) (c) (d) (e) (f)

Fig. 4: **ED success and failure. Success:** *given images (a) and (b) the cyan windows covering the bus and the aeroplane score high as the density of edges is concentrated in these regions.* **Failure:** *in image (c) the cyan window along with many other windows covering the water score high determining a high rate of false positives. In particular the windows covering the boats have a low score. We show the Canny edge maps in (d), (e) and (f).*

Superpixels segment an image into small regions of uniform color or texture. A key property of superpixels is to preserve object boundaries: all pixels in a superpixel belong to the same object [30]. Hence, an object is typically oversegmented into several superpixels, but none straddles its boundaries (fig. 5). Based on this property, we propose here a cue to estimate whether a window covers an object.

A superpixel s straddles a window w if it contains at least one pixel inside and at least one outside w. Most of the surface of an object window is covered by superpixels contained entirely inside it (w_1 in fig. 5c). Instead, most of the surface of a 'bad' window is covered by superpixels straddling it (i.e. superpixels continuing outside the window, w_2 in fig. 5c). The SS cue measures for all superpixels s the degree by which they straddle w:

$$\text{SS}(w, \theta_{SS}) = 1 - \sum_{s \in \mathcal{S}(\theta_{SS})} \frac{\min(|s \setminus w|, |s \bigcap w|)}{|w|} \tag{4}$$

where $\mathcal{S}(\theta_{SS})$ is the set of superpixels obtained using [11] with a segmentation scale θ_{SS}. For each superpixel s, eq. (4) computes its area $|s \bigcap w|$ inside w and its area $|s \setminus w|$ outside w. The minimum of the two is the degree by which s straddles w and is its contribution to the sum in eq. (4).

Superpixels entirely inside or outside w contribute 0 to the sum. For a straddling superpixel s, the contribution is lower when it is contained either mostly inside w, as part of the object, or mostly outside w, as part of the background (fig. 5c). Therefore, $\text{SS}(w, \theta_{SS})$ is highest for windows w fitting tightly around an object, as desired.

To the best of our knowledge, no earlier work has proposed a cue similar to SS. In sec. 5 we show that SS outperforms all other cues we consider (including MS and its original form [17]).

2.5. Implementation details.

MS. For every scale $s \in \{16, 24, 32, 48, 64\}$ and channel c we rescale the image to $s \times s$ and compute the score $\text{MS}(w, \theta_s)$ using one integral image [3].

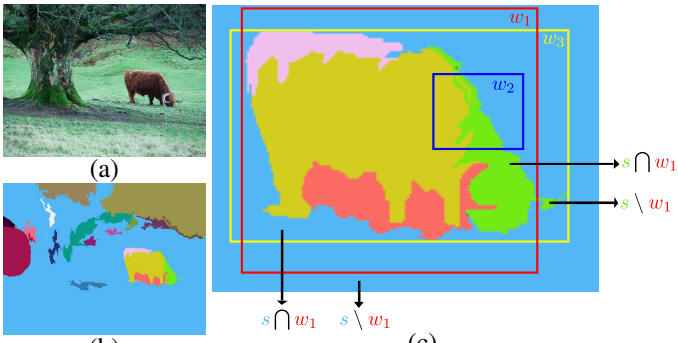


Fig. 5: **The SS cue.** *Given the segmentation (b) of image (a), for a window w we compute* $\mathrm{SS}(\mathrm{w}, \theta_{\mathrm{SS}})$ *(eq. 4). In (c), most of the surface of w_1 is covered by superpixels contained almost entirely inside it. Instead, all superpixels passing by w_2 continue largely outside it. Therefore, w_1 has a higher* SS *score than w_2. The window w_3 has an even higher score as it fits the object tightly.*

CC. We process the image in the quantized LAB space $8 \times 16 \times 16$ and compute $CC(w, \theta_{CC})$ using one integral images per quantized color.

ED. We rescale the image to 200×200 pixels and compute $ED(w, \theta_{ED})$ using one integral image.

SS. We obtain superpixels $\mathcal{S}(\theta_{SS})$ using the algorithm of [11] with segmentation scale θ_{SS}. We efficiently compute $\mathrm{SS}(w, \theta_{SS})$ using the following procedure. For each superpixel s we build an integral image $\mathrm{I^s(x, y)}$ giving the number of pixels of s in the rectangle $(0, 0) \rightarrow (x, y)$. Based on $\mathrm{I^s(x, y)}$, we can rapidly compute the number $|s \bigcap w|$ of pixels of s contained in any window w (fig. 6). The area $|s \setminus w|$ outside w is readily obtained as $|s \setminus w| = |s| - |s \bigcap w|$. Therefore, we can efficiently compute all elements of $\mathrm{SS}(w, \theta_{SS})$ (eq. 4).

3. Learning cue parameters

We learn the parameters of the objectness cues from a training dataset $\mathcal{T}$ consisting of 50 images randomly sampled from several well-known datasets, including INRIA Person [4], Pascal VOC 06 [8], and Caltech 101 [9].

These images contain a total of 291 instances of 48 diverse classes including a variety of objects such as sheep, buildings and keyboards. For training, we use annotated object windows $\mathcal{O}$, but not their class labels as our aim is to learn a measure *generic over classes*.

There are 8 parameters to be learned: θ_{CC}, θ_{ED}, θ_{SS} and θ^s_{MS} (for 5 scales s). We can learn from a rather small training set thanks to the low number of parameters and to the ability to produce many training examples from every image (fig. 8).

CC, ED, SS. We learn θ_{CC}, θ_{ED} and θ_{SS} in a Bayesian framework. As all three are learned in the same manner, we restrict the explanation to $\theta = \theta_{CC}$. For every image in $\mathcal{T}$ we generate 100000 random windows uniformly distributed over the entire image. Windows covering [2] an annotated object are considered positive examples ($\mathcal{W}^{\mathrm{obj}}$), the others negative ($\mathcal{W}^{\mathrm{bg}}$) (fig. 8).

[2] We follow the widespread PASCAL criterion [7], and consider a window w to cover an object o if $|w \bigcap o| / |w \bigcup o| > 0.5$.

Fig. 6: **Efficiently computing eq. (4).** *Window w_2 from fig. 5c is straddled by three superpixels. The individual superpixel contributions* $\frac{\min(|s \setminus w_2|, |s \bigcap w_2|)}{|w_2|}$ *are computed based on the three integral images* $\mathrm{I^s}$. *We show the integral images* $\mathrm{I^s}$ *in gray value.*

For any value of θ we can build the likelihoods for the positive $p_\theta(\mathrm{CC}(w, \theta)|\mathrm{obj})$ and negative classes $p_\theta(\mathrm{CC}(w, \theta)|\mathrm{bg})$, as histograms over the positive/negative training windows. Note how these histograms depend on θ.

We now find the optimal θ^* by maximizing the posterior probability that object windows are classified as positives

$$\theta^* = \arg\max_\theta \prod_{w \in \mathcal{W}^{\mathrm{obj}}} p_\theta(\mathrm{obj}|\mathrm{CC}(w, \theta)) = \quad (5)$$
$$= \arg\max_\theta \prod_{w \in \mathcal{W}^{\mathrm{obj}}} \frac{p_\theta(\mathrm{CC}(w, \theta)|\mathrm{obj}) \cdot p(\mathrm{obj})}{\sum_{c \in \{\mathrm{obj}, \mathrm{bg}\}} p_\theta(\mathrm{CC}(w, \theta)|c) \cdot p(c)}$$

where the priors are set by relative frequency: $p(\mathrm{obj}) = |\mathcal{W}^{\mathrm{obj}}| / (|\mathcal{W}^{\mathrm{obj}}| + |\mathcal{W}^{\mathrm{bg}}|)$, $p(\mathrm{bg}) = 1 - p(\mathrm{obj})$.

The advantage of this procedure is that the distribution of training samples is close to what the method will see when a new test image is presented, as opposed to just using the ground-truth and a few negative samples. Moreover, it is likely to generalize well, as it is trained from many variants of the annotated windows in $\mathcal{W}^{\mathrm{obj}}$ (i.e. all those passing the PASCAL criterion, which is also how the method will be evaluated on test images, sec. 5).

For a window w, the learned parameters θ_{CC} and θ_{ED} define the outer ring $\mathrm{Surr}(w, \theta_{CC})$ and the inner ring $\mathrm{Inn}(w, \theta_{ED})$. The learned parameter θ_{SS} defines the superpixel segmentation scale.

MS. We learn each threshold θ^s_{MS} independently, by optimizing the localization accuracy of the training objects $\mathcal{O}$ at each scale s. For every training image I and scale s, we compute the saliency map I^s_{MS} and the MS score of every possible window. Running non-maximum suppression on this 4D score space [3] gives a set of local maxima windows $\mathcal{W}^{\max}_s$. We find the optimal $\theta^{s*}_{\mathrm{MS}}$ by maximizing

$$\theta^{s*}_{\mathrm{MS}} = \arg\max_{\theta^s_{\mathrm{MS}}} \sum_{o \in \mathcal{O}} \max_{w \in \mathcal{W}^{\max}_s} \frac{|w \bigcap o|}{|w \bigcup o|} \quad (6)$$

i.e. we seek for the threshold $\theta^{s*}_{\mathrm{MS}}$ that leads to local maxima of MS most accurately covering the annotated objects $\mathcal{O}$ (according to the PASCAL criterion). Notice how this procedure is discriminative, as maximizing (6) implicitly entails also minimizing the score of windows without annotated objects.

[3] This is implemented efficiently by the method of [29]

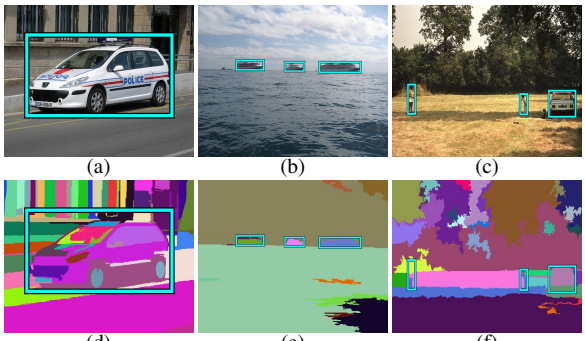


Fig. 7: **SS success and failure. Success**: *The cyan windows in (a) and (b) have high* SS *score computed from segmentations (d) and (e).* **Failure**: *Segmentation produces superpixels (f) not preserving the boundaries of the small objects in (c), resulting in low* SS.

4. Bayesian cue integration

Since the four cues are quite complementary, using several of them at the same time appears promising. MS gives only a rough indication of where an object is as it is designed to find blob-like things (fig. 2). Instead, CC provides more accurate windows, but sometimes misses objects entirely (fig. 3). ED provides many false positives on textured areas (fig. 4). SS is very distinctive but depends on good superpixels, which are fragile for small objects (fig. 7).

To combine cues we train a Bayesian classifier to distinguish between positive and negative quadruplets of values $(\mathrm{MS}, \mathrm{CC}, \mathrm{ED}, \mathrm{SS})$. For each training image, we sample 100000 windows from the distribution given by the MS cue (thus biasing towards better locations), and then compute the other cues for them. Windows covering an annotated object are considered as positive examples $\mathcal{W}^{\mathrm{obj}}$, all others as negative $\mathcal{W}^{\mathrm{bg}}$.

A natural way to combine our cues is to model them jointly. Unfortunately, it would require an enormous number of samples to estimate the joint likelihood $p(\mathrm{cue}_1, \ldots, \mathrm{cue}_n|\mathrm{obj})$, where $\mathrm{cue}_i \in \mathcal{C} = \{\mathrm{MS}, \mathrm{CC}, \mathrm{ED}, \mathrm{SS}\}$, $1 \leq i \leq n$, $n = 4$. Therefore, we choose a Naive Bayes approach here. We have also tried a linear discriminant, but it performed worse in our experiments, probably because it combines cues in a too simplistic manner (i.e. just a weighted sum).

In the Naive Bayes model, the cues are independent, so training consists of estimating the priors $p(\mathrm{obj}), p(\mathrm{bg})$, which we set by relative frequency, and the individual cue likelihoods $p(\mathrm{cue}|\mathrm{c})$, for $\mathrm{cue} \in \mathcal{C}$ and $c \in \{\mathrm{obj}, \mathrm{bg}\}$, from the large sets of training windows $\mathcal{W}^{\mathrm{obj}}, \mathcal{W}^{\mathrm{bg}}$.

After training, when a test image is given, we can sample any desired number T of windows from MS and then compute the other cues for them (as done above for obtaining training windows). Considering a subset of cues $\mathcal{A} \subseteq \mathcal{C}$, the posterior probability of one of these test windows w is

$$p(\mathrm{obj}|\mathcal{A}) = \frac{p(\mathcal{A}|\mathrm{obj})p(\mathrm{obj})}{p(\mathcal{A})} = \frac{p(\mathrm{obj}) \prod_{\mathrm{cue} \in \mathcal{A}} p(\mathrm{cue}|\mathrm{obj})}{\sum_{c \in \{\mathrm{obj}, \mathrm{bg}\}} p(\mathrm{c}) \prod_{\mathrm{cue} \in \mathcal{A}} p(\mathrm{cue}|\mathrm{c})} \quad (7)$$

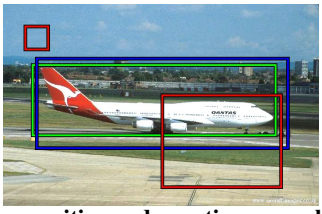


Fig. 8: **Obtaining positive and negative examples.** *Given an image we generate 100000 uniformly distributed windows and label them as positive or negative according to the PASCAL criterion. We show the annotated object window (green), a positive (blue) and two negative (red) examples.*

This posterior constitutes the final objectness score of w. The T test windows and their scores (7) form a distribution from which we can sample any desired final number F of windows. Note how eq. (7) allows us to combine any subset $\mathcal{A}$ of cues, e.g. pairs of cues $\mathcal{A} = \{\mathrm{MS}, \mathrm{CC}\}$, triplets $\mathcal{A} = \{\mathrm{MS}, \mathrm{CC}, \mathrm{SS}\}$ or all cues $\mathcal{A} = \mathcal{C} = \{\mathrm{MS}, \mathrm{CC}, \mathrm{ED}, \mathrm{SS}\}$. Function (7) can combine any subset rapidly without recomputing the likelihoods.

Sampling procedure. T window scores form a multinomial distribution D . Naively sampling F windows from D requires $T \cdot F$ operations, so we use an efficient sampling procedure. From the T scores we build the cumulative sum score vector v. Note how the elements of v are sorted in ascending order and the last vector element $v(T)$ is the sum of all scores. To sample a window we first generate a random number u uniformly distributed in $[0, v(\mathrm{T})]$. Then we do a binary search in v to retrieve the interval $[v_{i-1}, v_i]$ containing u. The chosen sample i has score v_i. Hence, sampling F windows only costs $\mathrm{F} \cdot \log_2 \mathrm{T}$ operations. We always use this procedure to sample from a multinomial in this paper.

5. Experiments

We evaluate the performance of our objectness measure on the popular **PASCAL VOC 07** dataset [7], which is commonly used to evaluate class-specific object detectors [4, 10, 32]. In **PASCAL VOC 07** each image is annotated with ground-truth bounding-boxes of objects from twenty categories (boat, bicycle, horse, *etc.*). Thus, it is very suitable for our evaluation, as we want to find *all* objects in the image, irrespective of their category. We evaluate on all 4952 images in the official test set [7] where objects appear against heavily cluttered backgrounds and vary greatly in location, scale, appearance, viewpoint and illumination.

Evaluation setup. As there are billions of windows in an image it is expensive to compute each cue for all windows. Moreover, our measure is intended as a focus of attention mechanism for later applications, so it is useful to output a reasonable number of windows likely to cover objects.

For SS, MS and ED, we build a distribution D by scoring all the T windows on a regular $4D$ grid. Computing CC is more expensive due to the large number of integral images involved, so we build a distribution D by scoring $T = 100000$ windows uniformly distributed over the image. Each cue is evaluated below on $F = 1000$ windows per image sampled from D.

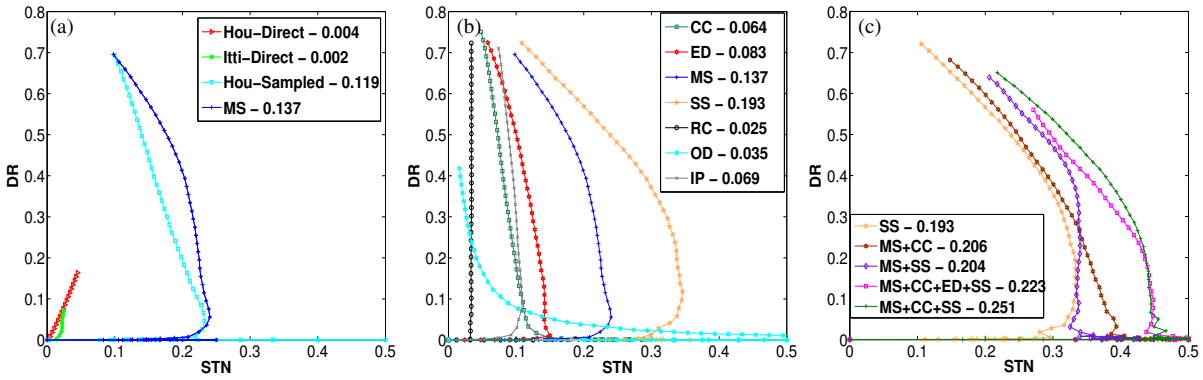


Fig. 9: **DR/STN plots.** *(a)* MS *vs [17, 19]; (b) single cues vs baselines; (c) cue combinations vs* SS*. We report the* ALC *for each curve.*

For a cue combination, to build D we sample $T = 100000$ windows from the distribution given by MS, compute the other cues for them and score them with eq. (7). Each cue combination is evaluated below on $F = 1000$ samples from D. Therefore, each single cue and cue combination is evaluated uniformly on 1000 windows per image.

DR-STN curves. The performance of a method is evaluated with detection-rate/signal-to-noise (DR/STN) curves. DR is the percentage of ground-truth objects covered by a window output by the method. An object is only considered covered by a window if the strict PASCAL-overall criterion [7] is satisfied (intersection-over-union > 0.5). STN is the percentage of windows covering a ground-truth object. We summarize a DR/STN curve with a single value: the area to the left of the curve (ALC), and use it to compare methods.

MS vs [17, 19] We compare our MS cue to [17] and [19] (fig. 9a). The Hou-Direct curve refers to [17] as originally proposed. A single saliency map at scale $s = 64$ is thresholded at its average intensity[4]. Each connected component in the resulting binary map is output as an object. The Itti-Direct curve refers to [19] as originally proposed. The saliency map is computed and the most salient objects are extracted from it using Itti's procedure [19][5].

Both procedures make hard decisions and output only few windows per image (about 5-10). As the curves show, both methods fail on the challenging PASCAL VOC 07 dataset. Hou-Direct performs better than Itti-Direct, but still reaches only about 0.18 DR and 0.08 STN.

For a fair comparison to our method, the Hou-Sampled curve reports performance when inserting [17] into our framework: we use their saliency map in our window score (eq. 1) and learn the threshold θ_{64} from our training set as in eq. (6). Finally, the MS curve uses all color channels and multiple scales, with separate scale-specific learned thresholds. DR/STN performance curves improve steadily as we progress towards our MS cue, showing that all the components we propose contribute to its success.

Baselines. In the following we compare our objectness measure to three baselines (fig 9b): random chance (RC), interest points (IP), and a histogram of oriented gradients (HOG) [4] detector (OD). For RC we generate 1000 random windows with random scores. For IP we extract interest points [27] and score every window in a 4D regular grid as $\text{IP}(w) = \frac{1}{\sqrt{|w|}} \sum_{\{p \in w\}} \text{cornerness}(p)$. For evaluation we sample 1000 windows from this grid according to their scores. HOG was shown to perform well in class-specific detection. Here we train it to detect objects of arbitrary classes: for OD we train a single HOG detector from all objects in the same 50 images used to train our method (sec. 3). From all detections, we sample 1000 windows according to their scores.

Single cues. Fig. 9b reports performance for our single cues and for the baselines. All our cues perform far above chance (RC). All cues but CC outperform IP, with MS and SS leading by a wide margin. This shows that finding entire objects cannot simply be reduced to interest point detection. Moreover, our newly proposed SS cue performs best, substantially above the second best cue MS (and therefore also above [17, 19]). This demonstrates SS is a powerful alternative to traditional 'stand out' saliency cues. Finally, note how OD performs very poorly, which confirms that generic object detection is different from class-specific detection. We believe OD fails because no single pattern of gradients within a window is characteristic for objects in general, whereas our cues are designed to capture this.

Cue combinations. Combining cues in our Bayesian framework improves results for all cue combinations but those including both SS and ED (fig. 9c). Adding CC to a combination of the best two single cues MS + SS further raises performance and leads to the best cue combination MS + CC + SS. This shows they are complementary cues, all necessary for finding objects in highly challenging images. Combining cues raises STN, as it makes the objectness measure more distinctive. 23% of the 1000 windows sampled from MS + CC + SS cover an object. This high STN is valuable for algorithms taking sampled objectness windows as input, such as class-specific object detectors (next section).

6. Speeding up class-specific detectors

Many state-of-the-art class-specific object detectors [5, 10, 15] are based on sliding windows. In sec. 6.1 we give a general algorithm for using our objectness measure as a lo-

[4] this gives better results than the threshold of [17], i.e. 3× the average

[5] using software fromhttp://www.saliencytoolbox.net/

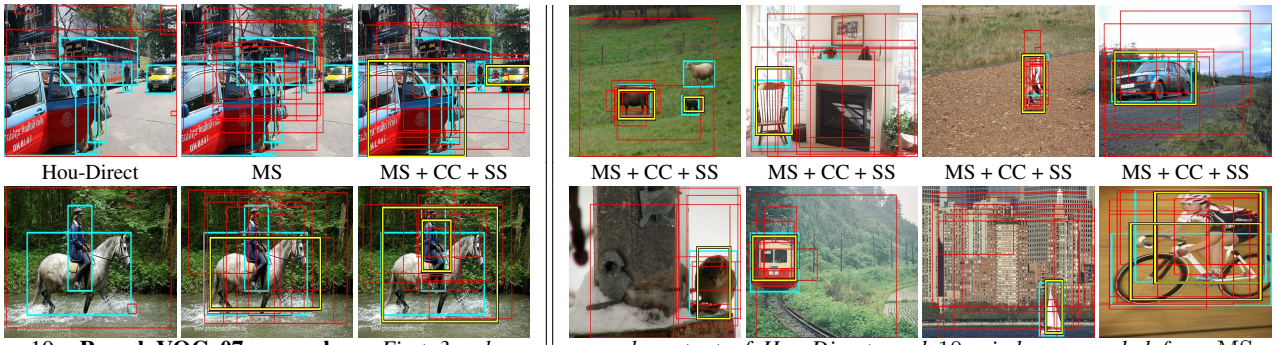


Fig. 10: **Pascal VOC 07 examples.** *First 3 columns: example output of Hou-Direct, and* 10 *windows sampled from* MS *and* MS + CC + SS. *We mark in yellow windows correctly covering ground-truth object (cyan); if there is more than one correct window, the best one is shown; all other windows are in red. Hou-Direct output windows rarely correctly cover an object. MS finds some of the objects, whereas MS + SS + CC is more accurate and finds more objects. The last four columns show* 10 *windows sampled from our final objectness measure* MS + CC + SS. *It can find many of the objects in these highly challenging images.*

Algorithm 1 Using objectness for class-specific detectors.

Input: F, D, c
Output: $\mathcal{D}et$
1: $\mathcal{I} = \{w_1, \ldots, w_F\}, w_i \rightsquigarrow D, \forall i$
2: $\mathcal{I}_s = \{(w_1, s_{w_1}), \ldots, (w_F, s_{w_F})\}, s_{w_i} = c(w_i), \forall i.$
3: $\mathcal{P}_s = NMS(\mathcal{I}_s) = \{(w_{n_1}, s_{w_{n_1}}), \ldots, (w_{n_P}, s_{w_{n_P}})\}$
4: $\mathcal{LM} = \{w^{lm}_{n_1}, \ldots, w^{lm}_{n_P}\}, w^{lm}_{n_j} = \max_{w \in \mathcal{V}_{w_{n_j}}} s_w$
5: $\mathcal{D}et = NMS(\mathcal{LM})$

cation prior for *any* sliding window detector. In sec. 6.2-6.4 we detail how this algorithm works for three particular detectors [4, 10, 21]. Using objectness greatly reduces the number of windows evaluated by the detectors (sec. 6.5).

6.1. General algorithm

The general scheme for using our objectness measure as a location prior for object detectors is algorithm 1. The algorithm inputs the class-specific confidence function c which the detector employs to score a window.

We build an initial set $\mathcal{I}$ of $\mathrm{F} = 1000$ windows sampled from the distribution D of windows scored by our objectness measure MS + CC + SS (line 1). We use c to score each window in $\mathcal{I}$ (line 2). We then run the non-maxima suppression of [10] to remove every window overlapping more than 50% with a higher scored window. This results in a set $\mathcal{P}_s$ of promising windows (line 3). For every window $w_p \in \mathcal{P}_s$, we iteratively move to the local maximum of c in its neighborhood $\mathcal{V}_{w_p}$, resulting in window w^{lm}_p (line 4). Finally, we run NMS on the local maxima windows $\mathcal{LM}$ and obtain detections $\mathcal{D}et$ (line 5).

In order to use this algorithm one has to specify a window scoring function c, which is specific to a particular detector and object class, and a window neighborhood $\mathcal{V}$.

6.2. Speeding up [4]

The detector of [4] models a class by a single window filter c on HOG features. In [4] this filter is applied at all positions (x, y) and scales s on a grid over image. In our algorithm instead, we only apply c to the sampled windows $w_i \in \mathcal{I}$. Next, we search for a local maximum in the neighborhood $\mathcal{V}_{w_i}$ on the grid, for up to 5 iterations.

6.3. Speeding up [10]

The detector of [10] models a class with a mixtures of multiscale deformable part models. More complex than [4], the model includes a root filter and a collection of part filters and associated deformation models. The score of a window at location (x, y, s) combines the score of the root filter, the parts and a deformation cost measuring the deviation of the part from its ideal location.

In our algorithm, we score only the sampled windows $w_i \in \mathcal{I}$ and apply the neighborhood search as in sec. 6.2.

6.4. Speeding up Bag-of-words detectors

The window scoring function c can also be a Bag-of-visual-words classifier [21], which represents a window with a histogram of local features quantized over a precomputed codebook. Here we follow the setup of [21] and use as c a linear SVM classifier.

We score only the sampled windows $w_i \in \mathcal{I}$ and set the neighborhood $\mathcal{V}_{w_i}$ to all windows obtained by translating w_i by 2 pixels and/or scaling w_i by factor 1.1. We iterate local search until convergence (usually about 10 iterations).

Since for this type of window scoring function it is possible to apply the branch-and-bound technique ESS [21], we compare to ESS rather than to traditional sliding-windows on a grid [4, 10].

6.5. Quantitative evaluation

We compare the 3 object detectors [4, 10, 21] to our algorithm 1 on the entire PASCAL VOC 07 test set (20 object classes over 4952 images) using the standard PASCAL VOC protocol [7].

Our algorithm uses the same window scoring function c used by the corresponding detector. We train it for [10] using their source code[6]. For [4] we train [10] with only the root filter (no parts). For [21] we obtained image features and SVM hyperplanes from the authors. For detection, we use the source code of [10] for both [10] and [4], and the source code of ESS [7] for [21].

[6] http://people.cs.uchicago.edu/~pff/latent/
[7] http://sites.google.com/site/christophlampert/software

Table 1: For each detector [4, 10, 21] we report its performance (left column) and that of our algorithm 1 using the same window scoring function (right column). We show the average number of windows evaluated per image #win and the detection performance as the mean average precision (mAP) over all 20 classes.

	[4]	OBJ- [4]	[10]	OBJ- [10]	ESS-BOW[21]	OBJ-BOW
mAP	0.186	0.161	0.263	0.224	0.127	0.112
#win	79945	2033	18562	1998	183501	2960

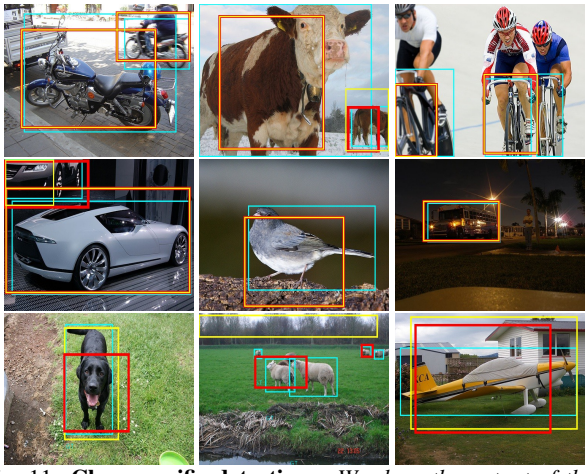


Fig. 11: **Class-specific detections.** *We show the output of three object detectors in yellow (first row - [4], second row - [10], third row -*ESS*). We show in red the output of our algorithm using the same window scoring function as the corresponding detector. Note how often the yellow and the red windows are identical. Ground-truth objects are in cyan.*

As tab. 1 shows, our algorithm evaluates $10x$-$40x$ fewer windows than sliding-windows [4, 10]. Interestingly, it also evaluates $50x$ fewer windows than ESS [21] (notice that the implicit search space of [21] is larger than that of [4, 10] as it contains all windows, not just a grid). Moreover, our algorithm's mAP is only slightly below the original detectors (-0.026 on average), showing this massive speedup [8] comes with little compromise on performance. Finally, our algorithm is general, as it supports *any* window scoring function. ESS instead requires a (tight) bound on the best score in a contiguous set of windows (for example, for the scoring functions of [4, 10] no bound is currently known and ESS is not applicable).

7. Conclusions

We presented an objectness measure trained to distinguish object windows from background ones. It combines several image cues, including the innovative SS cue. We demonstrated that this cue outperforms traditional saliency [17] and that the combined objectness measure performs better than any cue alone. Moreover, we have given an algorithm to employ objectness to greatly reduce the number of windows evaluated by class-specific detectors. Objectness can be useful also in other applications, such as to help learning object classes in a weakly supervised scenario, where object locations are unknown, and for tracking objects in video.

[8] The additional cost to compute objectness and sample 1000 windows $\mathcal{I}$ is negligible as it is done only once per image. The same windows $\mathcal{I}$ are reused for all 20 classes, and can in fact be reused for *any* class.

References

[1] J. Arpit, R. Saiprasad, and M. Anurag. Multi-stage contour based detection of deformable objects. In *ECCV*, 2008.
[2] N. Bruce and J. Tsotsos. Saliency based on information maximization. In *NIPS*, 2005.
[3] F. Crow. Summed-area tables for texture mapping. In *SIGGRAPH*, 1984.
[4] N. Dalal and B. Triggs. Histogram of Oriented Gradients for Human Detection. In *CVPR*, 2005.
[5] C. Desai, D. Ramanan, and C. Folkess. Discriminative models for multi-class object layout. In *ICCV*, 2009.
[6] T. Deselaers, B. Alexe, and V. Ferrari. Localizing objects while learning their appearance. Technical report, ETH Zurich, 2010.
[7] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007.
[8] M. Everingham, L. Van Gool, C. K. I. Williams, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2006.
[9] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In *CVPR Workshop of Generative Model Based Vision*, 2004.
[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2009. in press.
[11] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, Sep 2004.
[12] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.
[13] D. Gao and N. Vasconcelos. Bottom-up saliency is a discriminant process. In *ICCV*, 2007.
[14] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *NIPS*, 2007.
[15] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *ICCV*, 2009.
[16] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008.
[17] X. Hou and L. Zhang. Saliency detection: A spectral residual approach. In *CVPR*, 2007.
[18] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998.
[19] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *PAMI*, 20(11):1254–1259, 1998.
[20] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In *ECCV*, 2004.
[21] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.
[22] B. Leibe and B. Schiele. Scale-invariant object categorization using a scale-adaptive mean-shift search. In *DAGM*, 2004.
[23] T. Liu, J. Sun, N. Zheng, X. Tang, and H. Shum. Learning to detect a salient object. In *CVPR*, 2007.
[24] D. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, Sep 1999.
[25] Y. F. Ma and H. J. Zhang. Contrast-based image attention analysis by using fuzzy growing. In *ICMM*, 2003.
[26] L. Marchesotti, C. Cifarelli, G. Csurka. A framework for visual saliency detection with applications to image thumbnailing. In *ICCV*, 2009.
[27] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 1(60):63–86, 2004.
[28] F. Moosmann, D. Larlus, and F. Jurie. Learning saliency maps for object categorization. In *ECCV*, 2006.
[29] A. Neubeck and L. Van Gool. Efficient non-maximum suppression. In *ICPR*, 2006.
[30] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006.
[31] R. Valenti, N. Sebe, and T. Gevers. Image saliency by isocentric curvedness and color. In *ICCV*, 2009.
[32] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.
[33] G. Wang and D. Forsyth. Joint learning of visual attributes, object classes and visual saliency. In *ICCV*, 2009.
[34] J. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *ICCV*, pages 756–763, 2005.