

What makes for effective detection proposals?

Jan Hosang¹, Rodrigo Benenson¹, Piotr Dollár², and Bernt Schiele¹

¹Max Planck Institute for Informatics

²Facebook AI Research (FAIR)

Abstract—Current top performing object detectors employ detection proposals to guide the search for objects, thereby avoiding exhaustive sliding window search across images. Despite the popularity and widespread use of detection proposals, it is unclear which trade-offs are made when using them during object detection. We provide an in-depth analysis of twelve proposal methods along with four baselines regarding proposal repeatability, ground truth annotation recall on PASCAL and ImageNet, and impact on DPM and R-CNN detection performance. Our analysis shows that for object detection improving proposal localisation accuracy is as important as improving recall. We introduce a novel metric, the average recall (AR), which rewards both high recall and good localisation and correlates surprisingly well with detector performance. Our findings show common strengths and weaknesses of existing methods, and provide insights and metrics for selecting and tuning proposal methods.

I. INTRODUCTION

As recently as a few years ago, the most successful approaches to object detection utilised the well known “sliding window” paradigm [1]–[3], in which a computationally efficient classifier tests for object presence in every candidate image window. Sliding window classifiers scale linearly with the number of windows tested, and while single-scale detection requires classifying around 10^4 to 10^5 windows per image, the number of windows grows by an order of magnitude for multi-scale detection. Modern detection datasets [4], [5] also require the prediction of object aspect ratio, further increasing the search space to 10^6 to 10^7 windows per image.

The steady increase in complexity of the core classifiers has led to improved detection quality, but at the cost of significantly increased computation time per window [6]–[10]. One approach for overcoming the tension between computational tractability and high detection quality is through the use of “detection proposals” [11]–[14]. Under the assumption that all objects of interest share common visual properties that distinguish them from the background, one can design or train a method that, given an image, outputs a set of proposal regions that are likely to contain objects. If high object recall can be reached with considerably fewer windows than used by sliding window detectors, significant speed-ups can be achieved, enabling the use of more sophisticated classifiers.

Current top performing object detectors for PASCAL [4] and ImageNet [5] all use detection proposals [6]–[10]. In addition to allowing for use of more sophisticated classifiers, the use of detection proposals alters the data distribution that the classifiers handle. This may also improve detection quality by reducing spurious false positives.

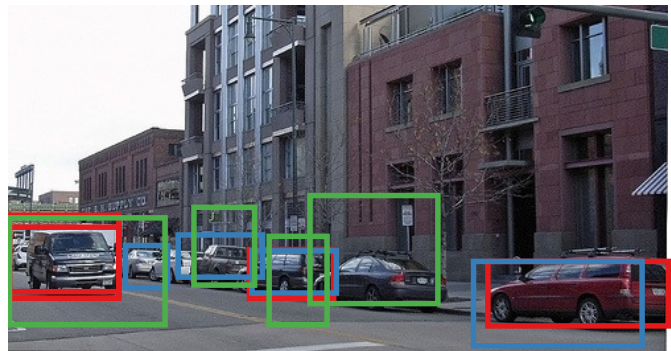


Figure 1: What makes object detection proposals effective?

Most papers on generating detection proposals perform fairly limited evaluations, comparing results using only a subset of metrics, datasets, and competing methods. In this work, we aim to revisit existing work on proposals and compare most publicly available methods in a unified framework. While this requires us to carefully re-examine the metrics and settings for evaluating proposals, it allows us to better understand the benefits and limitations of current methods.

The contributions of this work are as follows:

- In §II we provide a systematic overview of detection proposal methods and define simple baselines that serve as reference points. We discuss the taxonomy of proposal methods, and describe commonalities and differences of the various approaches.
- In §III we introduce the notion of proposal repeatability, discuss why it matters when considering proposals for object detection, and measure the repeatability of existing methods. The results are somewhat unexpected.
- In §IV we study object recall on the PASCAL VOC 2007 test set [4], and for the first time, over the larger and more diverse ImageNet 2013 val set [5]. The latter allows us to examine possible biases towards PASCAL object categories. Overall, these experiments are substantially broader in scope than previous work, both in the number of methods evaluated and datasets used.
- In §V we evaluate the influence of different proposal methods on DPM [3] and R-CNN [7] detection performance. Based on our results, we introduce a novel evaluation metric, the average recall (AR). We show that AR is highly correlated with detector performance, more so than previous metrics, and we advocate AR to become the standard metric for evaluating proposals. Our

experiments provide the first clear guidelines for selecting and tuning proposal methods for object detection.

All evaluation scripts and method bounding boxes used in this work are publicly available to facilitate the reproduction of our evaluation¹. The results presented in this paper summarise results of over 500 experiments on multiple data sets and required multiple months of CPU time.

An earlier version of this work appeared in [15].

II. DETECTION PROPOSAL METHODS

Detection proposals are similar in spirit to interest point detectors [28], [29]. Interest points allow for focusing attention to the most salient and distinctive locations in an image, greatly reducing computation for subsequent tasks such as classification, retrieval, matching, and detection. Likewise, object proposals considerably reduce computation compared to the dense (sliding window) detection framework by generating candidate proposals that may contain objects. This in turn enables use of expensive classifiers per window [6]–[10].

It is worthwhile noting that interest points were dominant when computing feature descriptors densely was prohibitive. However, with improved algorithmic efficiency and increased computational power, it is now standard practice to use dense feature extraction [30]. The opposite trend has occurred in object detection, where the dense sliding window framework has been overtaken by use of proposals. We aim to understand if detection proposals improve detection accuracy or if their use is strictly necessary for computational reasons. While in this work we focus on the impact of proposals on detection, proposals have applications beyond object detection, as we discuss in §VI.

Two general approaches for generating object proposals have emerged: *grouping methods* and *window scoring methods*. These are perhaps best exemplified by the early and well known *SelectiveSearch* [14] and *Objectness* [11] proposal methods. We survey these approaches in §II-A and §II-B, followed by an overview of alternate approaches in §II-C and baselines in §II-D. Finally, we consider the connection between proposals and cascades in §II-E and provide additional method details in §II-F.

The survey that follows is meant to be exhaustive. However, for the purpose of our evaluations, we only consider methods for which source code is available. We cover a diverse set of methods (in terms of quality, speed, and underlying approach). Table I gives an overview of the 12 selected methods (plus 4 baselines).² Table I also indicates high level information regarding the output of each method and a qualitative overview of the results of the evaluations performed in the remainder of this paper.

In this paper we concentrate on class-agnostic proposals for single-frame, bounding box detection. For proposal methods that output segmentations instead of bounding boxes, we convert the output to bounding boxes for the purpose of our evaluation. Methods that operate on videos and require temporal information (e.g. [31]) are considered outside the scope of this work.

A. Grouping proposal methods

Grouping proposal methods attempt to generate multiple (possibly overlapping) segments that are likely to correspond to objects. The simplest such approach would be to directly use the output of any hierarchical image segmentation algorithm, e.g. Gu et al. [32] use the segmentation produced by gPb [33]. To increase the number of candidate segments, most methods attempt to diversify such hierarchies, e.g. by using multiple low level segmentations [17], [24], [27] or starting with an over-segmentation and randomising the merge process [24]. The decision to merge segments is typically based on a diverse set of cues including superpixel shape, appearance cues, and boundary estimates (typically obtained from [33], [34]).

We classify grouping methods into three types according to how they generate proposals. Broadly speaking, methods generate region proposals by grouping superpixels (SP), often using [35], solving multiple graph cut (GC) problems with diverse seeds, or directly from edge contours (EC), e.g. from [33], [34], [36]. In the method descriptions below the type of each method is marked by SP, GC, or EC accordingly.

We note that while all the grouping approaches have the strength of producing a segmentation mask of the object, we evaluate only the enclosing bounding box proposals.

- **SelectiveSearch**^{†SP} [14], [27] greedily merges superpixels to generate proposals. The method has no learned parameters, instead features and similarity functions for merging superpixels are manually designed. *SelectiveSearch* has been broadly used as the proposal method of choice by many state-of-the-art object detectors, including the R-CNN detector [7].
- **RandomizedPrim**^{†SP} [24] uses similar features as *SelectiveSearch*, but introduces a randomised superpixel merging process in which all probabilities have been learned. Speed is substantially improved.
- **Rantalankila**^{†SP} [25] proposes a superpixel merging strategy similar to *SelectiveSearch*, but using different features. In a subsequent stage, the generated segments are used as seeds for solving graph cuts in the spirit of CPMC (see below) to generate more proposals.
- **Chang**^{SP} [37] combine saliency and *Objectness* with a graphical model to merge superpixels into figure/background segmentations.
- **CPMC**^{†GC} [12], [17] avoids initial segmentations and computes graph cuts with several different seeds and unaries directly on pixels. The resulting segments are ranked using a large pool of features.
- **Endres**^{†GC} [13], [19] builds a hierarchical segmentation from occlusion boundaries and solves graph cuts with different seeds and parameters to generate segments. The proposals are ranked based on a wide range of cues and in a way that encourages diversity.
- **Rigor**^{†GC} [26] is a somewhat improved variant of CPMC that speeds computation considerably by re-using computation across multiple graph-cut problems and using the fast edge detectors from [34], [38].
- **Geodesic**^{†EC} [20] starts from an over-segmentation of the image based on [34]. Classifiers are used to place

¹Project page: <http://goo.gl/uMhkAs>

²We mark the evaluated methods with a ‘†’ in the following listing.

Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repeatability	Recall Results	Detection Results
Bing [16]	Window scoring		✓	✓	0.2	***	*	.
CPMC [17]	Grouping	✓	✓		250	-	**	*
EdgeBoxes [18]	Window scoring		✓	✓	0.3	**	***	**
Endres [19]	Grouping	✓	✓	✓	100	-	**	**
Geodesic [20]	Grouping	✓		✓	1	*	***	**
MCG [21]	Grouping	✓	✓		30	*	***	**
Objectness [22]	Window scoring		✓	✓	3	.	.	.
Rahtu [23]	Window scoring		✓	✓	3	.	.	*
RandomizedPrim's [24]	Grouping	✓		✓	1	*	*	*
Rantalankila [25]	Grouping	✓		✓	10	**	.	*
Rigor [26]	Grouping	✓		✓	10	*	**	**
SelectiveSearch [27]	Grouping	✓	✓	✓	10	**	***	**
Gaussian				✓	0	.	.	*
SlidingWindow				✓	0	***	.	.
Superpixels		✓			1	*	.	.
Uniform				✓	0	.	.	.

Table I: Comparison of different detection proposal methods. Grey check-marks indicate that the number of proposals is controlled by indirectly adjusting parameters. Repeatability, quality, and detection rankings are provided as rough summary of the experimental results: “-” indicates no data, “.”, “*”, “**”, “***” indicate progressively better results. These guidelines were obtained based on experiments presented in sections §III, §IV, and §V, respectively.

seeds for a geodesic distance transform. Level sets of each of the distance transforms define the object/background segmentations that are the proposals.

- **MCG^{†EC}** [21] introduces a fast algorithm for computing multi-scale hierarchical segmentations building on [33], [34], [36]. Segments are merged based on edge strength and the resulting object hypotheses are ranked using cues such as size, location, shape, and edge strength.

B. Window scoring proposal methods

An alternate approach for generating detection proposals is to score each candidate window according to how likely it is to contain an object of interest. Compared to grouping approaches these methods usually only return bounding boxes and tend to be faster. Unless window sampling is performed very densely, this approach typically generates proposals with low localisation accuracy. Some methods counteract this by refining the location of the generated windows.

- **Objectness[†]** [11], [22] is one of the earliest and well known proposal methods. An initial set of proposals is selected from salient locations in an image, these proposals are then scored according to multiple cues including colour, edges, location, size, and the strong “superpixel straddling” cue.
- **Rahtu[†]** [23] begins with a large pool of proposal regions generated from individual superpixels, pairs and triplets of superpixels, and multiple randomly sampled boxes. The scoring strategy used by *Objectness* is revisited, and major improvements are proposed. [39] adds additional low-level features and highlights the importance of a properly tuned non-maximum suppression.
- **Bing[†]** [16] uses a simple linear classifier trained over edge features and applied in a sliding window manner. Using adequate approximations a very fast class agnostic detector is obtained (1 ms/image on CPU). However, it was shown that the classifier has minimal influence and similar performance can be obtained *without* looking at

the image pixels [40]. This image independent method is named *CrackingBing*.

- **EdgeBoxes^{†EC}** [18] also starts from a coarse sliding window pattern, but builds on object boundary estimates (obtained via structured decision forests [34], [41]) and adds a subsequent refinement step to improve localisation. No parameters are learned. The authors propose tuning the density of the sliding window pattern and the threshold of the non-maximum suppression to tune the method for different overlap thresholds (see §V).
- **Feng** [42] poses proposal generation as the search for salient image content and introduces new saliency measures, including the ease with which a potential object can be composed from the rest of the image. The sliding window paradigm is used and every location scored according to the saliency cues.
- **Zhang** [43] proposes a method that trains a cascade of ranking SVMs on simple gradient features. The first stage has different classifiers for different scales and aspect ratios; the second stage ranks all proposals from the previous stage. All SVMs are trained using structured output learning to score windows higher that overlap more with objects. Because the cascade is trained and tested over the same set of categories, it is unclear how well this approach generalises across categories.
- **RandomizedSeeds** [44] uses multiple randomised SEED superpixel maps [45] to score each candidate window. The scoring is done using a simple metric similar to “superpixel straddling” from *Objectness*, no additional cues are used. The authors show that using multiple superpixel maps significantly improves recall.

C. Alternate proposal methods

- **ShapeSharing** [46] is a non-parametric, data-driven method that transfers object shapes from exemplars into test images by matching edges. The resulting regions are subsequently merged and refined by solving graph cuts.

- **Multibox** [8], [47] trains a neural network to directly regress a fixed number of proposals in the image without sliding the network over the image. Each of the proposals has its own location bias to diversify the location of the proposals. The authors report top results on ImageNet.

D. Baseline proposal methods

We additionally consider a set of baseline methods that serve as reference points. Just as all the evaluated methods described earlier, the following baselines are class independent:

- **Uniform**[†]: To generate proposals, we uniformly sample the bounding box centre position, square root area, and log aspect ratio. We estimate the range of these parameters on the PASCAL VOC 2007 training set after discarding 0.5% of the smallest and largest values, so that our estimated distribution covers 99% of the data.
- **Gaussian**[†]: Likewise, we estimate a multivariate Gaussian distribution for the bounding box centre position, square root area, and log aspect ratio. After calculating mean and covariance on the training set we sample proposals from this distribution.
- **SlidingWindow**[†]: We place windows on a regular grid as is common for sliding window object detectors. The requested number of proposals is distributed across window sizes (width and height), and for each window size, we place the windows uniformly. This procedure is inspired by the implementation of Bing [16], [40].
- **Superpixels**[†]: As we will show, superpixels have an important influence on the behaviour of the proposal methods. Since five of the evaluated methods build on [35], we use it as a baseline: each low-level segment is used as a detection proposal. This method serves as a lower-bound on recall for methods using superpixels.

It should be noted that with the exception of Superpixels, all the baselines generate proposal windows independent of the image content. SlidingWindow is deterministic given the image size (similar to CrackingBing), while the Uniform and Gaussian baselines are stochastic.

E. Proposals versus cascades

Many proposal methods utilise image features to generate candidate windows. One can interpret this process as a discriminative one; given such features a method quickly determines whether a window should be considered for detection. Indeed, many of the surveyed methods include some form of discriminative learning (SelectiveSearch and EdgeBoxes are notable exceptions). As such, proposal methods are related to cascades [2], [48]–[50], which use a fast but inaccurate classifier to discard a vast majority of unpromising proposals. Although traditionally used for class specific detection, cascades can also apply to sets of categories [51], [52].

The key distinction between traditional cascades and proposal methods is that the latter is required to generalise beyond object classes observed during training. So what allows discriminatively trained proposal methods to generalise to unseen categories? A key assumption is that training a classifier for a

large enough number of categories is sufficient to generalise to unseen categories (for example, after training on cats and dogs proposals may generalise to other animals). Additionally, the discriminative power of the classifier is often limited (e.g. Bing and Zhang), thus preventing overfitting to the training classes and forcing the classifier to learn coarse properties shared by all object (e.g. “objects are roundish”).

F. Controlling the number of proposals

In the following sections we will perform an extensive apples-to-apples comparison of the 12 methods (plus 4 baselines) listed in table I. In order to be able to compare amongst methods, for each method we need to control the number of proposals produced per image. By default, the evaluated methods provide variable numbers of detection proposals, ranging from just a few ($\sim 10^2$) to a large number ($\sim 10^5$). Additionally, some methods output sorted or scored proposals, while others do not. Having more proposals increases the chance for high recall, thus for each method in all experiments we attempt to carefully control the number of generated proposals. Details are provided next.

Objectness, CPMC, Endres, Selective Search, Rahtu, Bing, MCG, and EdgeBoxes provide scored or sorted proposals, we use the top k . Rantalankila, Rigor, and Geodesic provide neither direct control over the number of proposals nor sorted proposals, but indirect control over k can be obtained by altering other parameters. Thus, we record the number of produced proposals on a subset of the images for different parameters and linearly interpolate between the parameter settings to control k . For RandomizedPrim, which lacks any control over the number of proposals, we randomly sample k proposals.

III. PROPOSAL REPEATABILITY

Training a detector on detection proposals rather than on all sliding windows modifies the distribution of both positive and negative windows. In section IV, we look into how well the different object proposals overlap with ground truth annotations of objects, which is an analysis of the positive window distribution. In this section we analyse the distribution of negative windows: if the proposal method does not consistently propose windows on similar image content without objects or with partial objects, the classifier may have difficulty generating scores on negative windows on the test set.

We call the property of proposals being placed on similar image content the *repeatability* of a proposal method. Intuitively the proposals should be repeatable on slightly different images with the same image content. To evaluate repeatability we project proposals from one image into another slightly modified image. The PASCAL VOC [4] does not contain suitable images. An alternative is the dataset of [29], but it only consists of 54 images and even fewer objects. Instead, we opt to apply synthetic transformations to PASCAL images.

A. Evaluation protocol for repeatability

Our evaluation protocol is inspired by [29], which evaluates interest point repeatability. For each image in the PASCAL

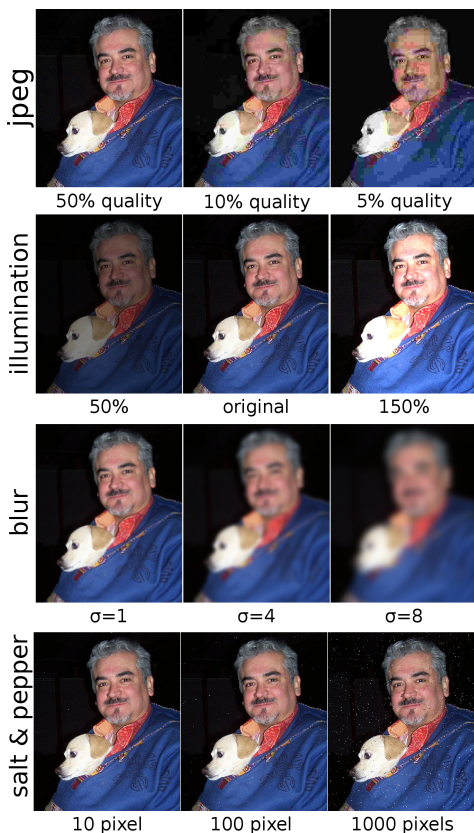


Figure 2: Illustration of the perturbation ranges used for repeatability experiments.

VOC 2007 test set [4], we generate several perturbed versions. We consider changes in scale, blur, rotation, illumination, JPEG compression, and “salt and pepper” noise (see figure 4).

For each pair of reference and perturbed images we compute detection proposals with a given algorithm (requesting 1000 windows per image). The proposals are projected back from the perturbed into the reference image and then matched to the proposals in the reference image. In the case of rotation, all proposals whose centre lies outside the image after projection are removed before matching. Matching is done greedily according to the intersection over union (IoU) criterion. Given the matching, we plot the recall for every IoU threshold and *define the repeatability to be the area under this “recall versus IoU threshold” curve between IoU 0 and 1*³. Methods that propose windows at similar locations at high IoU are more repeatable, since the area under the curve is larger.

One issue regarding such proposal matching is that large windows are more likely to match than smaller ones since the same perturbation will have a larger relative effect on smaller windows. This effect is important to consider since different methods have very different distributions of proposal window sizes as can be seen in figure 5a. To reduce the impact of this effect, we bin the original image windows by area into 10 groups, and evaluate the area under the recall versus IoU curve per size group. In figure 5b we show the

³In contrast to the average recall (AR) used in later sections, we use the area under the entire curve here. We are interested in how much proposals change, which is independent of the PASCAL overlap criterion.

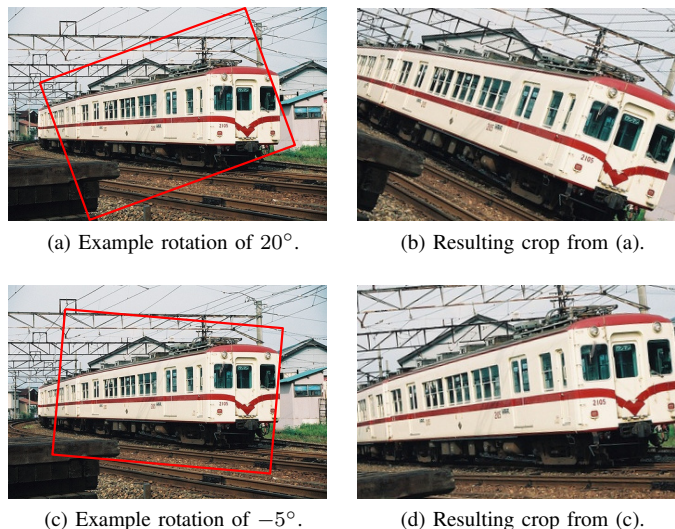


Figure 3: Examples of the rotation perturbation. (a) shows the largest rectangle with the same aspect as the original image that can fit into the image under a 20° rotation, and (b) the resulting crop. All other rotations are cropped to the same dimensions, e.g. the -5° rotation in (c) to the crop in (d).

recall versus IoU curve for a small blur perturbation for each of the 10 groups. As expected, large proposals have higher repeatability. In order to measure repeatability independent of the distribution of windows sizes for a given method, in all remaining repeatability experiments in figure 5 we show the (unweighted) average across the 10 size groups.

We omit the slowest two methods, CPMC and Endres, due to computational constraints (these experiments require running the detectors ~ 50 times on the entire PASCAL test set, once for every perturbation).

B. Repeatability experiments and results

There are some salient aspects of the result curves in figure 5 that need additional explanation. First, not all methods have 100% repeatability when there is no perturbation. This is due to random components in the selection of proposals for several methods. Attempting to remove a method’s random component is beyond the scope of this work and could potentially considerably alter the method. A second important aspect is the large drop of repeatability for most methods, even for very subtle image changes. We observed that many of the methods based on superpixels are particularly prone to such small perturbations. Indeed the Superpixels baseline itself shows high sensitivity to perturbations, so the instability of the superpixels themselves likely explains much of this effect. Inversely we notice that methods that are not based on superpixels are most robust to small image changes (e.g. Bing and also the baselines that ignore image content).

We now discuss the details and effects of each perturbation on repeatability, shown in figure 5:

Scale (5c): We uniformly sample the scale factor from $.5\times$ to $2\times$, and test additional scales near the original resolution



Figure 4: Example of the image perturbations considered. Top to bottom, left to right: original, then blur, illumination, JPEG artefact, rotation, scale perturbations, and “salt and pepper” noise.

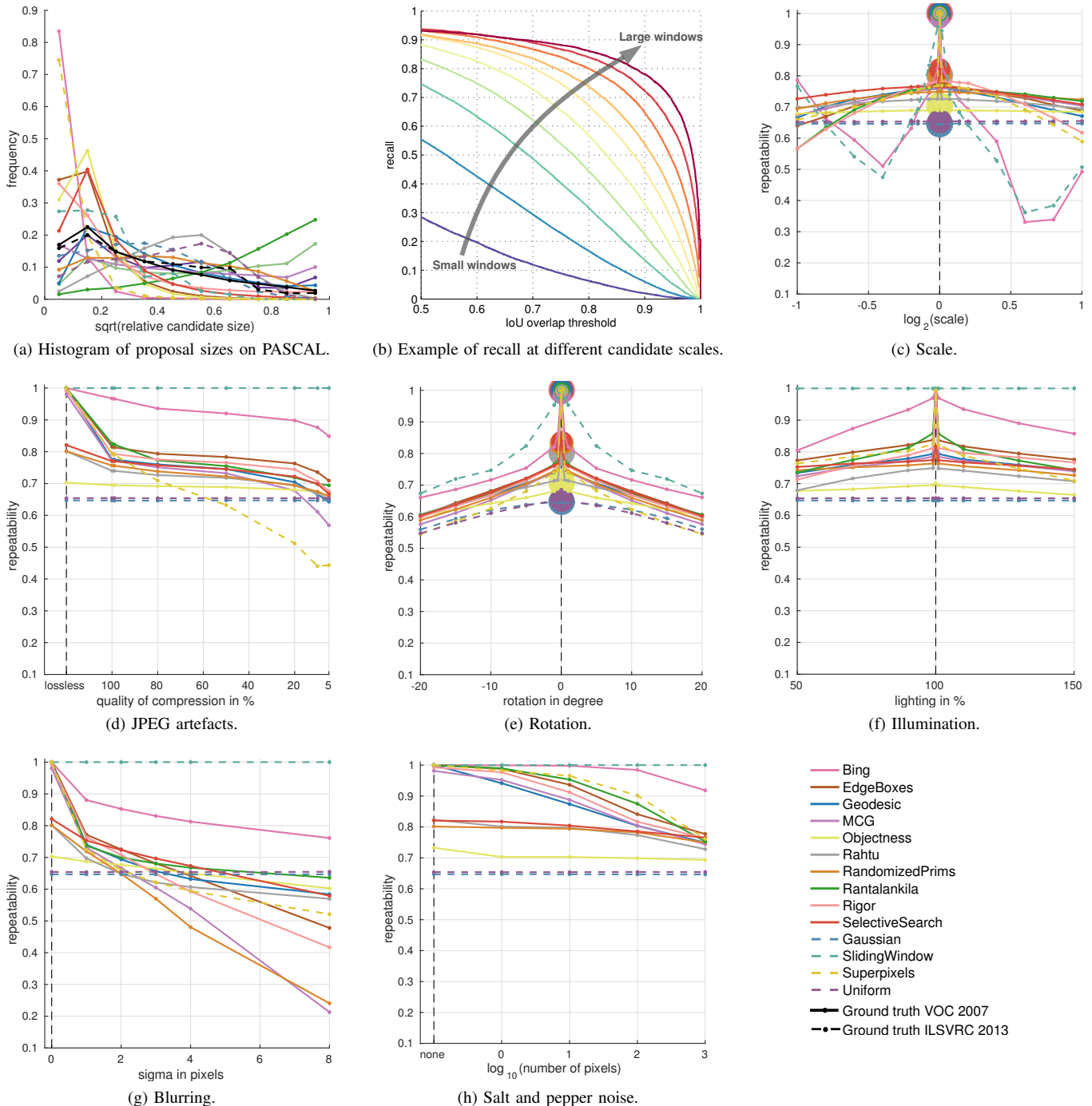


Figure 5: Repeatability results under various perturbations.

(.9 \times , .95 \times , .99 \times , 1.01 \times , 1.05 \times , 1.1 \times). Upscaling is done with bicubic interpolation and downscaling with anti-aliasing.

All methods except `Bing` show a drastic drop with small scale changes, but suffer only minor degradation for larger changes. `Bing` is more robust to small scale changes; however, it is more sensitive to larger changes due to its use of a coarse set of box sizes while searching for candidates (this also accounts for its dip in repeatability at half scales). The `SlidingWindow` baseline suffers from the same effect.

JPEG artefacts (5d): To create JPEG artefacts we write the target image to disk with the Matlab function `imwrite` and specify a quality settings ranging from 5% to 100%, see figure 2. Even the 100% quality setting is lossy, so we also include a lossless setting for comparison.

Similar to scale change, even slight compression has a large effect and more aggressive compression shows monotonic degradation. Despite using gradient information, `Bing` is most robust to these kind of changes.

Rotation (5e): We rotate the image in 5° steps between -20° and 20° . To ensure roughly the same content is visible under all rotations, we construct the largest box with the same aspect as the original image that fits into the image under a 20° rotation and use this crop for all other rotations, see figure 3.

All proposal methods are equally affected by image rotation. The drop of the `Uniform` and `Gaussian` baselines indicate the repeatability loss due to the fact that we are matching rotated bounding boxes.

Illumination (5f): To synthetically alter illumination of an image we changed its brightness channel in HSB colour space using `Imagemagick`. We vary the brightness between 50% and 150% of the original image so that some over and under saturation occurs, see figure 2.

Repeatability under illumination changes shows a similar trend as under JPEG artefacts. Methods based on superpixels are heavily affected. `Bing` is more robust, likely due to use of gradient information which is known to be fairly robust to illumination changes.

Blurring (5g): We blur the images with a `Gaussian` kernel with standard deviations $0 \leq \sigma \leq 8$, see figure 2.

The repeatability results again exhibit a similar trend although the drop is stronger for a small σ .

Salt and pepper noise (5h): We sample between 1 and 1000 random locations in the image and change the colour of the pixel to white if it is a dark and to black otherwise, see figure 2.

Surprisingly, most methods already lose some repeatability when even a single pixel is changed. Significant degradation in repeatability for the majority of the methods occurs when merely ten pixels are modified.

Discussion: Overall it seems that `Bing` and `EdgeBoxes` are more repeatable than other methods, possibly because both use machine learning components (SVM classifier for scoring and decision forests for features computation, respectively). Another reason of `Bing`'s repeatability is that its sliding window pattern has been designed to cover almost all possible

annotations with $\text{IoU} = 0.5$ (see also `Cracking Bing` [40]). We also conclude that the sensitivity of superpixels to image perturbations is a major cause for degradation in repeatability of several detection proposal methods.

Different applications will be more or less sensitive to repeatability. Our results indicate that if repeatability is a concern, the proposal method should be selected with care. For object detection, another aspect of interest is recall, which we explore in the next section.

IV. PROPOSAL RECALL

When using detection proposals for detection it is important to have a good coverage of the objects of interest in the test image, since missed objects cannot be recovered in the subsequent classification stage. Thus it is common practice to evaluate the quality of proposals based on the recall of the ground truth annotations.

A. Evaluation protocol for recall

The protocol introduced in [11] (using the PASCAL VOC 2007 dataset [4]) has served as a guideline for most evaluations in the literature. While previous papers do show various comparisons on PASCAL, the train and test sets vary amongst papers, and the metrics shown tend to favour different methods. We provide an extensive and unified evaluation and show that different metrics result in different rankings of proposal methods (e.g. see figure 6b versus 7b).

Metrics: Evaluating (class agnostic) detection proposals is quite different from traditional class-specific detection [53] since most metrics (class confusion, background confusion, precision, etc.) do not apply. Instead, one of the primary metrics for evaluating proposals is, for a fixed number of proposals, the fraction of ground truth annotations covered as the intersection over union (IoU) threshold is varied (figure 6). Another common and complementary metric is, for a fixed IoU threshold, the proposal recall as the number of proposals is varied (figure 7a, 7b). Finally, we define and report a novel metric, the average recall (AR) between IoU 0.5 to 1, and plot AR for varying number of proposals (figure 7c).

PASCAL: We evaluate recall on the full PASCAL VOC 2007 test set [4], which includes 20 object categories present in ~ 5000 unconstrained images. For the purpose of proposal evaluation we include all 20 object categories and all ground truth bounding boxes, including ‘‘difficult’’ ones, since our goal is to measure maximum recall. In contrast to [11], we compute a matching between proposals and ground truth, so one proposal cannot cover two objects. Note that while different methods may be trained on different sets of object categories and subsets of data, we believe evaluating on all categories at test time is appropriate as we care about absolute proposal quality. Such an evaluation strategy is further supported as many methods have no training stage, yet provide competitive results (e.g. `SelectiveSearch`).

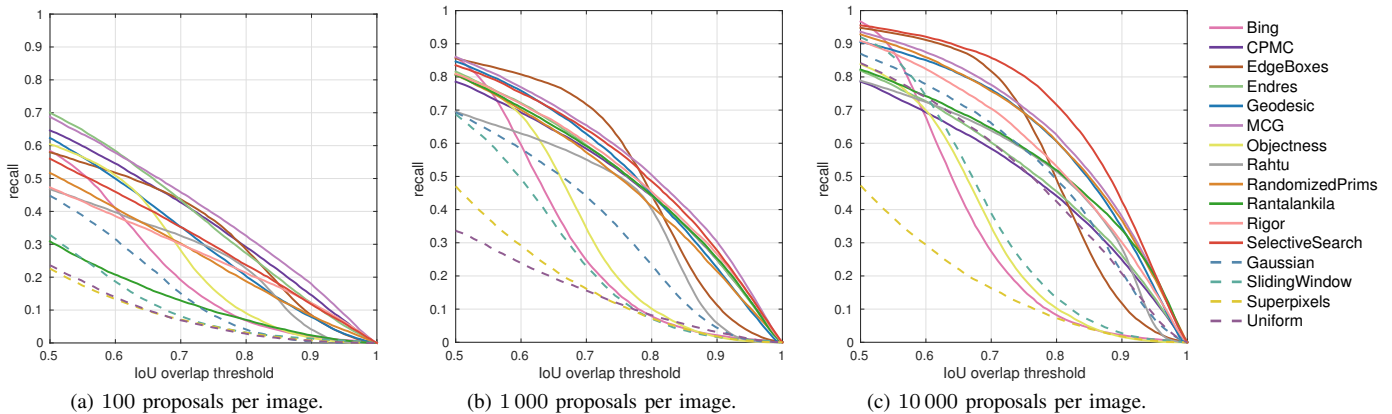


Figure 6: Recall versus IoU threshold on the PASCAL VOC 2007 test set.

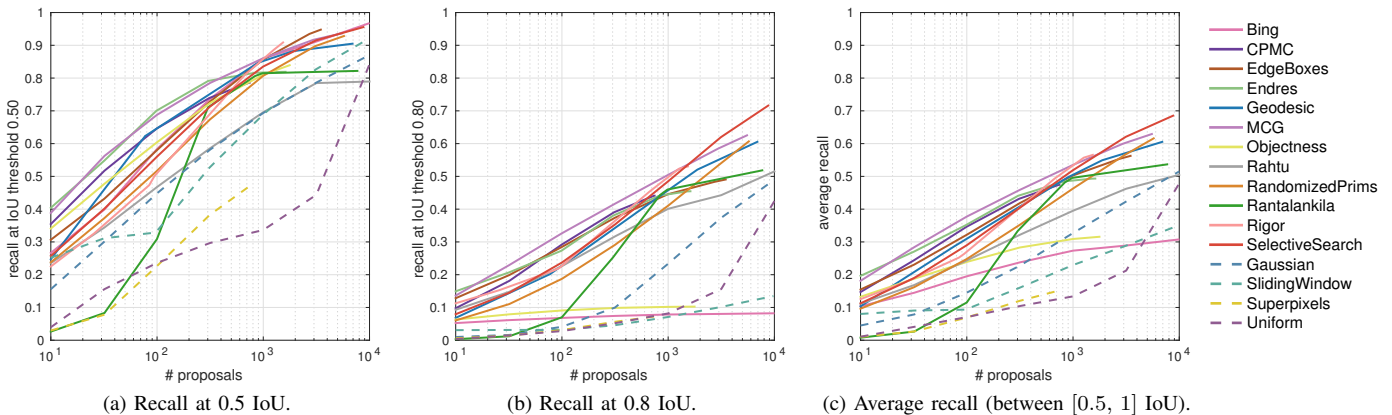


Figure 7: Recall versus number of proposal windows on the PASCAL VOC 2007 test set.

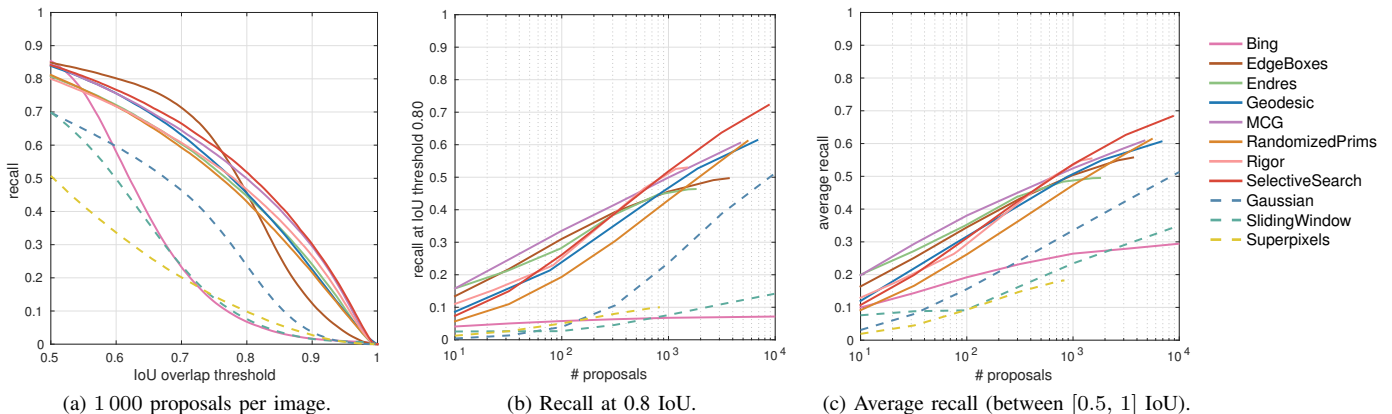


Figure 8: Recall on the ImageNet 2013 validation set.

ImageNet: The PASCAL VOC 2007 test set, on which most proposal methods have been previously evaluated, has only 20 categories, yet detection proposal methods claim to predict proposals for *any* object category. Thus there is some concern that the proposal methods may be tuned to the PASCAL categories and not generalise well to novel categories. To investigate this potential bias, we also evaluate methods on the larger ImageNet [5] 2013 validation set, which contains annotations for 200 categories in over $\sim 20\,000$ images. It should be noted that these 200 categories are *not* fine grained versions of the PASCAL ones. They include additional types of animals

(e.g. crustaceans), food items (e.g. hot-dogs), household items (e.g. diapers), and other diverse object categories.

B. Recall results

Results in figure 6 and 7 present a consistent trend across the different metrics. MCG, EdgeBoxes, SelectiveSearch, Rigor, and Geodesic are the best methods across different numbers of proposals. SelectiveSearch is surprisingly effective despite being fully hand-crafted (no machine learning involved). When considering less than 10^3 proposals, MCG, Endres, and CPMC provide strong results.

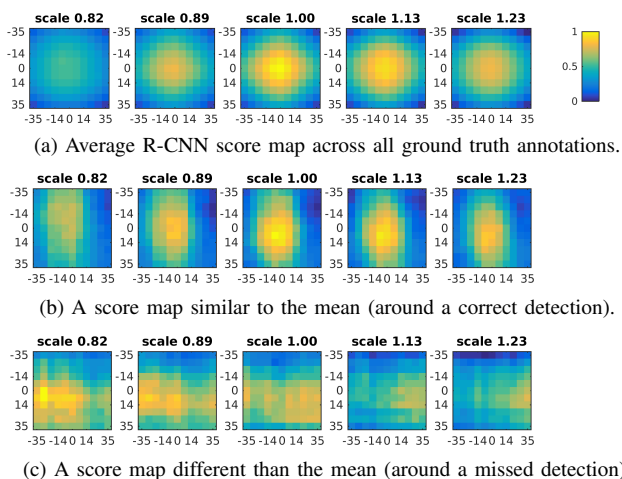


Figure 9: Normalised score maps of the R-CNN around ground truth annotations on the PASCAL 2007 test set. One grid cell in each map has width and height of ~ 7 px after the object height has been resized to the detector window of 227×227 px (3% of the object height).

Overall, the methods fall into two groups: well localised methods that gradually lose recall as the IoU threshold increases and methods that only provide coarse bounding box locations, so their recall drops rapidly. All baseline methods, as well as Bing, Rahtu, Objectness, and EdgeBoxes fall into the latter category. Bing in particular, while providing high repeatability, only provides high recall at $\text{IoU} = 0.5$ and drops dramatically when requiring higher overlap (the reason for this is identified in [40]).

Baselines: When inspecting figure 6 from left to right, one notices that with few proposals the baselines provide relatively low recall (figure 6a). However as the number of proposals increases, the Gaussian and SlidingWindow baselines become more competitive (figure 6b), and, given sufficient proposals, even the simple Uniform baseline performs reasonably (figure 6c). In relative gain, detection proposal methods have most to offer for low numbers of windows.

Average Recall: Rather than reporting recall at particular IoU thresholds, we also report the average recall (AR) between IoU 0.5 to 1, and plot AR for varying number of proposals in figure 7c. Much like the average precision (AP) metric for (class specific) object detection, AR summarises proposal performance across IoU thresholds (for a given number of proposals). In fact, in §V we will show that AR correlates well with detection performance. As can be seen in figure 7c, MCG performs well across the entire range of number of proposals. Endres and EdgeBoxes work well for a low number of proposals while for a higher number of proposals Rigor and SelectiveSearch perform best.

ImageNet: Figure 8 presents the results over ImageNet. As discussed, compared to PASCAL, ImageNet includes $10 \times$ ground truth classes and $4 \times$ images. Somewhat surprisingly the results are almost identical to the ones in figures 6b, 7b, and 7c. To understand this phenomenon, we note that the

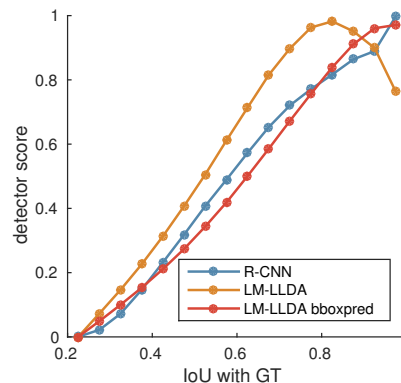


Figure 10: Normalised detector scores as a function of the overlap between the detector window and the ground truth.

statistics of ImageNet match the ones of PASCAL. In particular the typical image size and the mean number of object annotation per image (three) is identical in both datasets. This helps explaining why the recall behaviour is similar, and why methods trained on PASCAL still perform well on ImageNet.

Generalisation: We emphasise that although the results on PASCAL and ImageNet are quite similar, ImageNet covers 200 object categories, many of them unrelated to the 20 PASCAL categories. In other words, there is no measurable over-fitting of the proposal detection methods towards the PASCAL categories. This suggests that proposal methods transfer adequately amongst object classes, and can thus be considered true “objectness” measures.

V. USING THE DETECTION PROPOSALS

This section analyses detection proposals for use with object detectors. We consider two well known and quite distinct approaches to object detection. First we use a variant of the popular DPM part-based sliding window detector [3], specifically the LM-LLDA detector [54]. We also test the state of the art R-CNN [7] object detector which couples object proposals with a convolutional neural network classification stage. Our goals are twofold. First, we aim to measure the performance of different proposal methods for object detection. Second, we are interested in evaluating how well the proposal metrics reported in the previous sections can serve as a proxy for predicting final detection performance. For all experiments in this section we use 1000 proposals per method.

A. Detector responses around objects

As a preliminary experiment, we aim to quantify the importance of having high accuracy proposals. To gain intuition about how detectors are affected by proposal localisation, we build statistics of how detector scores change as a function of the overlap between the detector window and the ground truth annotation on the PASCAL 2007 test set [4].

Score map: Figure 9a shows the average R-CNN detection score around the ground truth annotations. We notice that the score map is symmetric and attains a maximum at the ground truth object location, in other words there is no systematic

Method	mAP	Method	mAP
LM-LLDA	33.5/34.4	SelectiveSearch	54.6
Bing	21.8/22.4	Bing	36.7
CPMC	29.9/30.7	CPMC	51.6
EdgeBoxes	31.8/32.2	EdgeBoxes	53.0
Endres	31.2/31.6	Endres	52.4
Geodesic	31.7/32.0	Geodesic	53.5
MCG	32.4/32.7	MCG	54.9
Objectness	25.0/25.4	Objectness	39.7
Rahtu	29.6/30.4	Rahtu	46.1
RandomizedPrims	30.5/30.9	RandomizedPrims	51.6
Rantalankila	30.7/31.3	Rantalankila	52.6
Rigor	31.5/32.1	Rigor	54.1
SelectiveSearch	31.7/32.3	SelectiveSearch	54.6
Gaussian	27.3/28.0	Gaussian	40.6
SlidingWindow	20.7/21.5	SlidingWindow	32.7
Superpixels	11.2/11.3	Superpixels	17.6
Uniform	16.6/16.9	Uniform	19.8

(a) LM-LLDA detection results (before / after bounding box regression). (b) R-CNN detection results (trained with SelectiveSearch).

Table II: Detection performance on PASCAL 2007 with different proposals for the LM-LLDA and R-CNN detectors.

spatial or scale bias in the detector. However, averaging the score maps removes small details and imperfections on individual score maps and results in a smooth and centred response. When considering individual activations (instead of the average), we observe a high variance in the quality of the score maps, as illustrated in figures 9b and 9c.

Score vs IoU: In figure 10 we show averaged detection scores as a function of their IoU overlap with the ground truth. The scores have been scaled between zero and one per class before averaging across classes. The drop of the LM-LLDA score at high overlaps is due to a location bias introduced during training by the latent location estimation on positive samples; this bias is compensated for by the subsequent bounding box prediction stage of LM-LLDA.

Localisation: We observe from figure 10 that both LM-LLDA and R-CNN exhibit an almost linear increase of detection score as IoU increases. From this we conclude that there is no IoU threshold that is “good enough” for obtaining top detection quality. We thus expect that improving localisation of proposals is as important as increasing ground truth recall.

B. LM-LLDA detection performance

We use pre-trained LM-LLDA [54] models to generate dense detections using the standard sliding window setup and subsequently apply different proposals to filter these detections at test time. This does not speed-up detection, but enables evaluating the effect of proposals on detection quality. A priori we may expect that detection results will deteriorate due to lost recall, but conversely, they may improve if the proposals filter out windows that would otherwise be false positives.

Implementation details: We take the raw detections of LM-LLDA before non-maximum suppression and filter them with the detection proposals of each method. We keep all detections that overlap more than 0.8 IoU with a candidate proposal; the

surviving detections are then non-maxima suppressed. As a final step we do bounding box regression, as is common for DPM models [3]. Note that with this procedure the detector also evaluates locations around each proposal window.

Results: Table IIa shows that using 1 000 detection proposals decreases detection quality compared with the original sliding window setup⁴ by about 1-2 mAP for the best performing proposal methods, see top row (LM-LLDA) versus the rows below. The five top performing methods all have mAP between 32.0 and 33.0 and are marked in bold: MCG, SelectiveSearch, EdgeBoxes, Rigor, and Geodesic. Note that the difference between these methods and the Gaussian baseline is relatively small (32.7 versus 28.0 mAP).

When we compare these results with figure 7c at 1000 proposals, we see that methods are ranked similarly. Methods with high average recall (AR) also have high mAP, and methods with lower AR also have lower mAP. The correlation between AR and mAP is analysed more closely in §V-D.

From table III we see that the per-class performance can be grouped into three cases: classes on which the best proposals (1) clearly hurt performance (bicycle, boat, bottle, car, chair, horse, mbike, person), (2) improve performance (cat, table, dog), (3) do not show significant change (all remaining classes). In the case of (1) we observe both reduced recall and reduced precision in the detection curves, probably because bad localisation decreases the scores of strong detections. There does not seem to be a correlation between poor LM-LLDA performance and the impact of proposal filtering.

C. R-CNN detection performance

The R-CNN object detector [7] couples detection proposals with a convolutional neural network classification stage. Because the classification stage of R-CNN is relatively slow per window, an exhaustive classification over all windows is prohibitive. Thus, in contrast to LM-LLDA, R-CNN cannot be run in a sliding window fashion and there is no sliding window baseline for R-CNN.

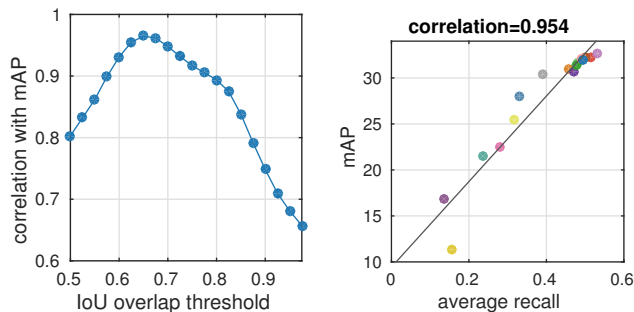
The R-CNN detector was trained and optimised for SelectiveSearch proposals. In principle, using the same proposals for training and testing is optimal. Unfortunately, retraining the R-CNN with each possible proposal method is not feasible. Instead, for our experiments we opt to use R-CNN trained with proposals generated by SelectiveSearch and only replace the proposals at test time. To estimate the effect of this simplification we retrained the R-CNN using EdgeBoxes. When using EdgeBoxes at test time, this improved results from 53.3 to 56.5 mAP.

Results: Although the absolute mAP numbers are considerably higher for R-CNN (an improvement of over 20 mAP), table IIb shows a similar trend as table IIa. As expected SelectiveSearch provides excellent results as the R-CNN was trained with SelectiveSearch, but multiple other methods get similar results. The five top performing methods (with mAP between 53.0 and 55.0) are the same as in the

⁴Not to be confused with the SlidingWindow proposals baseline.

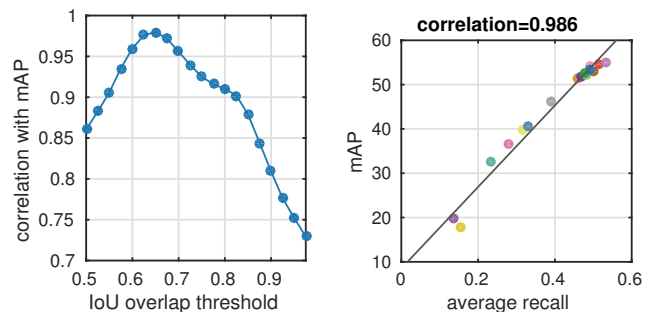
	aero	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	Avg.
LM-LLDA	33.7	61.3	12.4	18.5	26.7	53.0	57.2	22.4	22.7	25.6	25.1	14.0	59.2	51.0	39.1	13.6	21.7	38.0	48.8	44.0	34.4
Bing	-7.5	-23.2	-6.2	-8.1	-10.6	-13.3	-17.5	-6.8	-9.8	-15.4	-7.5	-1.4	-19.6	-19.0	-16.1	-3.4	-6.6	-18.1	-18.8	-10.0	-11.9
CPMC	-1.0	-15.0	-0.2	-4.4	-13.5	-1.8	-9.2	3.2	-9.1	-2.6	5.1	2.2	-4.2	-4.8	-7.0	-2.0	-2.6	1.2	-4.1	-4.9	-3.7
EdgeBoxes	-2.0	-6.1	-0.7	-3.8	-6.7	0.6	-5.8	-1.1	-2.0	-1.8	-4.6	0.4	-1.3	-1.3	-3.0	-1.7	-0.1	-0.9	-0.2	-1.1	-2.2
Endres	-1.4	-5.9	-0.5	-4.8	-13.3	-1.6	-7.3	3.6	-7.0	-3.2	4.9	1.9	-2.6	-2.5	-7.9	-2.8	-1.7	1.7	-0.5	-4.2	-2.8
Geodesic	-1.9	-8.4	-0.1	-4.5	-14.3	0.1	-6.7	2.5	-7.4	-1.3	4.4	2.5	-2.5	-3.3	-4.9	-1.1	-1.7	-0.1	1.2	0.0	-2.4
MCG	0.1	-8.8	0.0	-3.3	-7.1	-1.2	-8.8	3.7	-4.2	-2.0	5.4	2.7	-4.5	-1.8	-2.1	-2.0	-1.7	1.6	0.6	-0.7	-1.7
Objectness	-10.3	-15.1	-2.0	-6.2	-11.0	-9.5	-13.0	-3.6	-10.0	-6.4	-7.8	-1.0	-11.6	-15.9	-13.0	-2.7	-5.8	-11.2	-10.9	-12.9	-9.0
Rahtu	-0.3	-13.2	-0.3	-1.2	-13.0	-0.6	-12.0	3.3	-10.5	-4.3	2.0	2.1	-3.2	-4.9	-7.9	-2.8	-4.9	-5.0	0.0	-3.7	-4.0
Rand.Prim	2.1	-10.4	-0.5	-4.5	-13.2	-1.9	-10.1	5.0	-6.7	-3.5	2.0	2.4	-4.4	-5.1	-10.0	-2.3	-1.8	1.2	-3.8	-4.4	-3.5
Rantalankila	0.4	-13.5	0.4	-3.0	-13.5	-3.9	-9.0	4.6	-5.5	-3.6	3.9	2.8	-2.7	-4.0	-8.1	-2.5	-3.6	2.4	-1.5	-1.7	-3.1
Rigor	1.7	-7.9	0.5	-4.1	-12.4	-0.8	-9.0	6.3	-6.9	-1.7	1.8	2.9	-0.9	-3.3	-7.7	-1.8	-1.3	1.6	-1.2	-1.7	-2.3
Sel. Search	1.3	-7.7	1.0	-4.3	-11.1	-1.7	-7.8	3.9	-4.8	-1.5	5.4	2.2	-1.4	-3.8	-6.0	-1.5	-0.8	0.6	-2.4	-2.1	-2.1
Gaussian	-6.6	-13.4	-0.7	-4.4	-15.0	-6.1	-16.0	0.9	-9.1	-8.0	0.3	1.2	-4.2	-6.9	-10.3	-2.3	-6.5	-4.5	-3.6	-12.1	-6.4
SlidingWindow	-21.8	-20.7	-3.2	-8.1	-16.6	-14.7	-22.1	-0.7	-9.8	-11.7	-10.2	-1.4	-14.7	-20.1	-14.8	-3.8	-7.7	-21.0	-20.8	-14.8	-12.9
Superpixels	-23.9	-52.2	-3.1	-9.4	-17.4	-43.9	-42.3	-10.2	-11.3	-12.6	-15.8	-8.5	-50.1	-41.7	-30.9	-4.4	-10.6	-25.2	-39.7	-8.2	-23.1
Uniform	-7.6	-34.4	-6.3	-9.2	-17.6	-17.6	-26.2	-11.2	-13.3	-16.1	-5.7	-8.2	-35.7	-27.3	-25.9	-8.0	-12.5	-15.7	-18.7	-33.1	-17.5
Top methods avg.	-0.2	-7.8	0.1	-4.0	-10.3	-0.6	-7.6	3.1	-5.1	-1.7	2.5	2.2	-2.1	-2.7	-4.7	-1.6	-1.1	0.6	-0.4	-1.1	-2.1

Table III: Detection results on PASCAL VOC 2007. The top row indicates the average precision (AP) of LM-LLDA alone, while the other rows show the difference in AP when adding the proposal methods. Green indicates improvement of at least 2 AP, blue indicates no significant change ($-2 \leq \text{AP} < 2$), and white indicates performance decreases by more than 2 AP. EdgeBoxes achieves top results on 6 of the 20 categories; MCG performs best overall with -1.7 mAP loss.



(a) Correlation between mAP and re-call at different IoU thresholds. (b) Correlation between mAP and the average recall.

Figure 11: Correlation between detector performance of the LM-LLDA on PASCAL 07 and different proposal statistics.



(a) Correlation between mAP and re-call at different IoU thresholds. (b) Correlation between mAP and the average recall.

Figure 12: Correlation between detector performance of the R-CNN on PASCAL 07 and different proposal statistics.

LM-LLDA experiment. The gap between the top performance and the Gaussian baseline is more pronounced (54.9 versus 40.6 mAP), but the baseline still gets surprisingly good results considering that it disregards the image content.

D. Correlation between mAP and recall

To determine which of the recall statistics from section IV (figures 6 and 7) serves as the best predictor for detector performance, we quantify the observations from the previous subsections. In figures 11 and 12 we show the Pearson correlation coefficient between the detector performance and two recall statistics: recall at different overlap thresholds (figures 11a and 12a) and the *average recall* (AR) between 0.5 and 1 IoU⁵ (figures 11b and 12b). As before, for these experiments we use 1000 proposals per method.

⁵We compute the average between 0.5 and 1 IoU (and not between 0 and 1 as in §III), because we are not interested in recall below the PASCAL evaluation criterion of 0.5 IoU. Proposals with worse overlap than 0.5 are not only harder to classify correctly, but also would require a subsequent location refinement to become a successful detection.

We begin by examining the correlation between detection performance and recall at various IoU thresholds in figures 11a and 12a. We see that both detectors show a strong correlation (> 0.9) at an IoU range of roughly 0.6 to 0.8. Note that recall at IoU of 0.5 is actually only weakly correlated with detection performance, and methods that optimise for IoU of 0.5, such as Bing, are not well suited for generating proposals for object detection (see tables IIa and IIb). Thus, although recall at IoU of 0.5 has been traditionally used to evaluate object proposals, our analysis shows that is not a good metric for predicting detection performance.

The correlation between detection performance and AR is very strong, see figures 11b and 12b. Computing the AR over a partial IoU range (e.g. 0.6 to 0.8) further increases the correlation slightly; however, since the effect is minor, we opted to use AR over the entire range from 0.5 to 1.0 for simplicity. While the strong correlation does not imply that the AR can perfectly predict detection performance, as figures 11b and 12b show, the relationship is surprisingly linear.

We conclude that AR allows us to identify good proposal

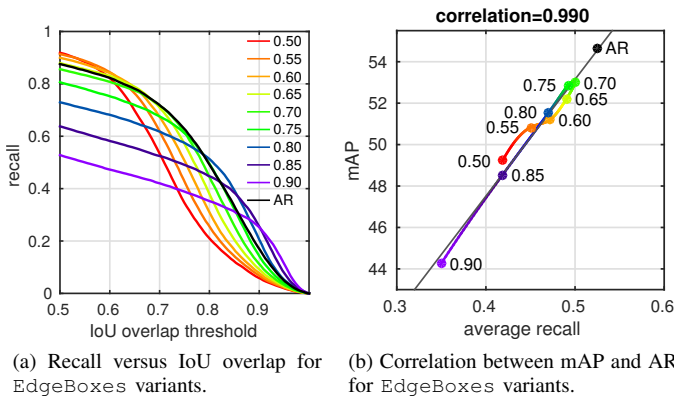


Figure 13: R-CNN detection performance after fine-tuning EdgeBoxes to maximise recall at different IoU values Δ , or to maximise average recall (AR); see §V-E for details. Maximising AR provides the best detection accuracy.

methods for object detection. The AR metric is easy to understand, easy to justify, is strongly correlated with detection performance, and is already included in our analysis of previous sections, see figure 7c. Overall we conclude that AR is the best predictor of detection performance; we suggest that future proposal methods should aim to optimise this metric.

E. Tuning proposal methods for mAP

In all previous experiments we have evaluated proposal methods using original parameter settings. However many methods have free parameters that allow for fine-tuning. For example, when adjusting window sampling density and the non-maximum suppression in EdgeBoxes [18], it is possible to trade-off low recall with good localisation for higher recall with worse localisation (a similar observation was made in [39]). In figure 13 we compare different versions of EdgeBoxes tuned to maximise recall at different IoU points Δ (we set $\alpha = \max(.65, \Delta - .05)$, $\beta = \Delta + .05$, see [18] for details). EdgeBoxes tuned for IoU of $\Delta = 0.70$ or 0.75 maximises AR and also results in the best R-CNN detection performance.

While the original EdgeBoxes method allows for optimising recall for a particular IoU threshold, we consider a new EdgeBoxes variant that directly maximises AR (marked "AR" in figure 13) to further explore the link between AR and detection quality. To do so, we alter its greedy non-maximum suppression (nms) procedure to make it *adaptive*. We start with a large nms threshold β_0 to encourage dense sampling around the top scoring candidates (a window is suppressed if its IoU with a higher scoring window exceeds this threshold). After selecting each proposal, β_{k+1} is slightly decreased via $\beta_{k+1} = \beta_k \cdot \eta$ to encourage greater proposal diversity. Setting $\beta_0 = 0.90$ and $\eta = 0.9996$ gave best AR at 1 000 proposals on the PASCAL validation set (we kept $\alpha = .65$ fixed). This new adaptive variant is not optimal at any particular IoU threshold, but has best overall AR and improves R-CNN mAP by 1.7 over the best previous EdgeBoxes variant (reaching mAP 54.7).

The results in figure 13 support our conclusion that the AR is a good predictor for mAP and suggest that AR can be used

for fine-tuning proposal methods. We expect other methods to improve as well if optimised for AR instead of a particular IoU threshold.

VI. DISCUSSION

In this work we have revisited the majority of existing detection proposal methods, proposed new evaluation metrics, and performed an extensive and direct comparison of existing methods. Our primary goal has been to enable practitioners to make more informed decisions when considering use of detection proposals and selecting the optimal proposal method for a given scenario. Additionally, our open source benchmark will enable more complete and informative evaluations in future research on detection proposals. We conclude by summarising our key findings and suggesting avenues for future work.

We found that the *repeatability* of virtually all proposal methods is limited: imperceptibly small changes to an image cause a noticeable change in the set of produced proposals. Even changing a single image pixel already exhibits measurable differences in repeatability. We foresee room for improvement by using more robust superpixel (or boundary estimation) methods. However, the importance of repeatability is application dependent. For object detection some robustness is desirable, but full repeatability is not the goal. Image independent methods such as SlidingWindow and CrackingBing have perfect repeatability but are inadequate for detection. Methods such as SelectiveSearch and EdgeBoxes seem to strike a better balance between recall and repeatability. We suspect that high quality proposal methods that are also more repeatable would yield improved detection accuracy, however, this has not been verified experimentally.

Our analysis showed that for object detection improving proposal *localisation accuracy* (improved IoU) is as important as improving recall. Indeed, we demonstrated that the popular metric of recall at IoU of 0.5 is not predictive of detection accuracy. As far as we know, our experiments are the first to demonstrate this effect. Proposals with high recall but at low overlap are not effective for object detection.

To simultaneously measure both proposal recall and localisation accuracy, we report *average recall* (AR), which summarises the distribution of recall across a range of overlap thresholds. Average recall correlates surprisingly well with detector performance (both for LM-LLDA and R-CNN), penalising proposal methods that optimise for a particular overlap. Average recall proves to be an excellent predictor of detection performance both for comparing competing methods as well as tuning parameters of a specific method. We encourage future work to report average recall (as shown in figures 7c/8c) as the primary metric for evaluating proposals for object detection.

Amongst the methods evaluated, the ones that perform best for object detection are MCG, SelectiveSearch, Rigor, Geodesic, and EdgeBoxes. If fast proposals are required, EdgeBoxes and Geodesic provide a good compromise between speed and quality. Surprisingly, these top performing methods all achieve fairly similar performance on object detection even though they employ very different mechanisms for generating proposals. SelectiveSearch merges

superpixels, Rigor computes multiple graph cut segmentations, MCG generates hierarchical segmentations, Geodesic employs the geodesic distance transform, and EdgeBoxes scores windows based on edge content.

Critically, we measured no visible drop in recall when going from the 20 PASCAL categories to the 200 ImageNet categories. This is an encouraging result indicating that *current methods do indeed generalise to different unseen categories*, and as such can be considered “true objectness” methods.

Do object proposals improve detection quality or are they just a transition technology until we have sufficient computing power? Our experiments indicate that dense sliding window evaluation, when computationally feasible, outperforms detection with object proposals (table III). In this light, proposals can be seen as sophisticated form of detection cascades, and as such may eventually become unnecessary. Given enough computing power, one would expect that a dense evaluation of CNNs would further improve performance over R-CNNs.

On the other hand, there are uses for detection proposals beyond object detection cascades. For example, they can be used to handle unknown categories at test time, or to enable weakly supervised learning [55]–[57].

Finally, we observe that current proposal methods reach high recall while using features that are not utilised by detectors such as LM-LLDA and R-CNN (e.g. object boundaries and superpixels). Conversely, with the exception of Multibox [47], none of the proposal methods use CNN features. We expect some cross-pollination will occur in this space.

In the future, detection proposals will surely improve in repeatability, recall, localisation accuracy, and speed. Top-down reasoning will likely play a more central role as purely bottom-up processes have difficulty generating perfect object proposals. We may also see a tighter integration between proposals and the detector, and the segmentation mask generated by many proposal methods may play a more important role during detection. One thing is clear: progress has been rapid in this young field and we expect proposal methods to evolve quickly over the coming years.

REFERENCES

- [1] C. Papageorgiou and T. Poggio, “A trainable system for object detection,” *IJCV*, 2000.
- [2] P. Viola and M. Jones, “Robust real-time face detection,” in *IJCV*, 2004.
- [3] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part-based models,” *PAMI*, 2010.
- [4] M. Everingham, S. Eslami, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge – a retrospective,” *IJCV*, 2014.
- [5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database,” in *CVPR*, 2009.
- [6] X. Wang, M. Yang, S. Zhu, and Y. Lin, “Regionlets for generic object detection,” in *ICCV*, 2013.
- [7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *CVPR*, 2014.
- [8] C. Szegedy, S. Reed, D. Erhan, and D. Anguelov, “Scalable, high-quality object detection,” *arXiv*, 2014.
- [9] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” in *ECCV*, 2014.
- [10] R. G. Cinbis, J. Verbeek, and C. Schmid, “Segmentation driven object detection with fisher vectors,” in *ICCV*, 2013.
- [11] B. Alexe, T. Deselaers, and V. Ferrari, “What is an object?” in *CVPR*, June 2010.
- [12] J. Carreira and C. Sminchisescu, “Constrained parametric min-cuts for automatic object segmentation,” in *CVPR*, 2010.
- [13] I. Endres and D. Hoiem, “Category independent object proposals,” in *ECCV*, 2010.
- [14] K. van de Sande, J. Uijlings, T. Gevers, and A. Smeulders, “Segmentation as selective search for object recognition,” in *ICCV*, 2011.
- [15] J. Hosang, R. Benenson, and B. Schiele, “How good are detection proposals, really?” in *BMVC*. British Machine Vision Association, September 2014.
- [16] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. H. S. Torr, “BING: Binarized normed gradients for objectness estimation at 300fps,” in *CVPR*, 2014.
- [17] J. Carreira and C. Sminchisescu, “Cpmc: Automatic object segmentation using constrained parametric min-cuts,” *PAMI*, 2012.
- [18] C. Zitnick and P. Dollár, “Edge boxes: Locating object proposals from edges,” in *ECCV*, 2014.
- [19] I. Endres and D. Hoiem, “Category-independent object proposals with diverse ranking,” in *PAMI*, 2014.
- [20] P. Krähenbühl and V. Koltun, “Geodesic object proposals,” in *ECCV*, 2014.
- [21] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marqués, and J. Malik, “Multiscale combinatorial grouping,” in *CVPR*, 2014.
- [22] B. Alexe, T. Deselaers, and V. Ferrari, “Measuring the objectness of image windows,” *PAMI*, 2012.
- [23] E. Rahtu, J. Kannala, and M. Blaschko, “Learning a category independent object detection cascade,” in *ICCV*, 2011.
- [24] S. Manén, M. Guillaumin, and L. Van Gool, “Prime object proposals with randomized prim’s algorithm,” in *ICCV*, 2013.
- [25] P. Rantalankila, J. Kannala, and E. Rahtu, “Generating object segmentation proposals using global and local search,” in *CVPR*, 2014.
- [26] A. Humayun, F. Li, and J. M. Rehg, “Rigor: Recycling inference in graph cuts for generating object regions,” in *CVPR*, 2014.
- [27] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, “Selective search for object recognition,” *IJCV*, 2013.
- [28] T. Tuytelaars and K. Mikolajczyk, “Local invariant feature detectors: a survey,” *Foundations and Trends in Computer Graphics and Vision*, 2008.
- [29] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool, “A comparison of affine region detectors,” *IJCV*, 2005.
- [30] T. Tuytelaars, “Dense interest points,” in *CVPR*, 2010.
- [31] K. Fragkiadaki, P. A. Arbeláez, P. Felsen, and J. Malik, “Spatio-temporal moving object proposals,” *arXiv*, 2014.
- [32] C. Gu, J. Lim, P. Arbelaez, and J. Malik, “Recognition using regions,” in *CVPR*, 2009.
- [33] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, “Contour detection and hierarchical image segmentation,” *PAMI*, 2011.
- [34] P. Dollár and C. L. Zitnick, “Fast edge detection using structured forests,” *PAMI*, 2015.
- [35] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient graph-based image segmentation,” *IJCV*, vol. 59(2), pp. 167–181, 2004.
- [36] X. Ren and L. Bo, “Discriminatively trained sparse code gradients for contour detection,” in *NIPS*, 2012.
- [37] K.-Y. Chang, T.-L. Liu, H.-T. Chen, and S.-H. Lai, “Fusing generic objectness and visual saliency for salient object detection,” in *ICCV*, 2011.
- [38] J. Lim, C. L. Zitnick, and P. Dollár, “Sketch tokens: A learned mid-level representation for contour and object detection,” in *CVPR*, 2013.
- [39] M. Blaschko, J. Kannala, and E. Rahtu, “Non Maximal Suppression in Cascaded Ranking Models,” in *Scandinavian Conference on Image Analysis*, Espoo, Finland, Jun. 2013, pp. 408–419.
- [40] Q. Zhao, Z. Liu, and B. Yin, “Cracking bing and beyond,” in *BMVC*, 2014.
- [41] P. Dollár and C. L. Zitnick, “Structured forests for fast edge detection,” in *ICCV*, 2013.
- [42] J. Feng, Y. Wei, L. Tao, C. Zhang, and J. Sun, “Salient object detection by composition,” in *ICCV*, 2011.
- [43] Z. Zhang, J. Warrell, and P. H. S. Torr, “Proposal generation for object detection using cascaded ranking svms,” in *CVPR*, 2011.
- [44] M. Van Den Bergh, G. Roig, X. Boix, S. Manen, and L. Van Gool, “Online video seeds for temporal window objectness,” in *ICCV*, 2013.
- [45] M. Van den Bergh, X. Boix, G. Roig, and L. Van Gool, “Seeds: Superpixels extracted via energy-driven sampling,” *IJCV*, 2014.
- [46] J. Kim and K. Grauman, “Shape Sharing for Object Segmentation,” in *ECCV*, 2012.
- [47] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, “Scalable object detection using deep neural networks,” in *CVPR*, 2014.

- [48] L. Bourdev and J. Brandt, "Robust object detection via soft cascade," in *CVPR*, 2005.
- [49] H. Harzallah, F. Jurie, and C. Schmid, "Combining efficient object localization and image classification," in *ICCV*, 2009.
- [50] P. Dollár, R. Appel, and W. Kienzle, "Crosstalk cascades for frame-rate pedestrian detection," in *ECCV*, 2012.
- [51] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing visual features for multiclass and multiview object detection," *PAMI*, 2007.
- [52] P. Zehnder, E. Koller-Meier, and L. Van Gool, "An efficient shared multi-class detection cascade," in *BMVC*, 2008.
- [53] D. Hoiem, Y. Chodpathumwan, and Q. Dai, "Diagnosing Error in Object Detectors," in *ECCV*, 2012.
- [54] R. Girshick and J. Malik, "Training deformable part models with decorrelated features," in *ICCV*, 2013.
- [55] S. Vicente, C. Rother, and V. Kolmogorov, "Object cosegmentation," in *CVPR*, 2011.
- [56] M. Guillaumin, D. Kuttel, and V. Ferrari, "Imagenet auto-annotation with segmentation propagation," *IJCV*, 2014.
- [57] K. Tang, A. Joulin, L.-J. Li, and L. Fei-Fei, "Co-localization in real-world images," in *CVPR*, 2014.