Branch and Bound Strategies for Non-maximal Suppression in Object Detection

Matthew B. Blaschko

Visual Geometry Group
Department of Engineering Science
University of Oxford
United Kingdom

Abstract. In this work, we are concerned with the detection of multiple objects in an image. We demonstrate that typically applied objectives have the structure of a random field model, but that the energies resulting from non-maximal suppression terms lead to the maximization of a submodular function. This is in general a difficult problem to solve, which is made worse by the very large size of the output space. We make use of an optimal approximation result for this form of problem by employing a greedy algorithm that finds one detection at a time. We show that we can adopt a branch-and-bound strategy that efficiently explores the space of *all* subwindows to optimally detect single objects while incorporating pairwise energies resulting from previous detections. This leads to a series of inter-related branch-and-bound optimizations, which we characterize by several new theoretical results. We then show empirically that optimal branch-and-bound efficiency gains can be achieved by a simple strategy of reusing priority queues from previous detections, resulting in speedups of up to a factor of three on the PASCAL VOC data set as compared with serial application of branch-and-bound.

1 Introduction

Non-maximal suppression has been employed in many settings in vision and image processing. In image processing, objectives for edge and corner detection have been specified in terms of the eigenvalues of a matrix containing local oriented image statistics [12], while more recently general objectives for object detection have been trained discriminatively [28, 5, 21, 2, 4, 3, 9]. Often, an objective function specifies a property of interest in image coordinates, but it is the arg maximum of the objective rather than scalar values that is of importance. From this perspective, an ideal objective would place all its mass on the true location and give zero output elsewhere. In practice, this is rarely the case, and the function output consists instead of hills and valleys characterizing intermediate belief in the fitness of a given location. Discriminative training of detection models can lead to the need for non-maximal suppression as more confident detections will have higher peaks than less confident ones. Without non-maximal suppression the next best-scoring detections will almost certainly be located on the upper slope of the peak corresponding with the most confident detection, while other peaks may be ignored entirely. One may interpret this as maximizing the log-likelihood of the detections assuming that they are independent, while in fact there is a strong spatial dependence on the scores of the output.

Here, we interpret commonly applied non-maximal suppression strategies as the maximization of a random-field model in which energies describing the joint distribution of detections are included. This insight enables us to characterize in general terms the maximization problem, and to make use of existing theoretical results on maximizing submodular (minimizing supermodular) functions. As a result, we can adopt an efficient optimization strategy with strong approximation guarantees. This is of particular interest as maximizing a submodular function is in general NP-hard. The resulting optimization problem can be solved by a series of inter-related optimizations. Here, we follow Lampert et al. and approach the optimization using a branch-and-bound strategy that enables fast detections of typically tens of milliseconds on a standard desktop machine [19].

The branch-and-bound strategy we consider here is a best first search that makes use of a priority queue to manage which regions of the space of detections to explore. Furthermore, the inter-related optimizations resulting from branch-and-bound have a very benign structure in that each problem can use intermediate results stored in the priority queue by the previous optimization. We show empirically that, while reuse of these results does not always give an optimal increase in speed, that there is a very simple strategy for the selective reuse of intermediate results that does give optimal empirical performance. This is further illuminated by several theoretical results that motivate the strategy.

1.1 Related Work

Viola and Jones developed one of the best studied and widely used generic detection algorithms [28]. A key step in their algorithm can be interpreted as non-maximal suppression, in which they cluster highly overlapping detections and represent clusters by only one detection. Thus, peaks in the detection landscape are compressed to a single detection, suppressing other output.

A key question in such strategies is which metric to use when suppressing detections that are too close. A common approach in the recent object detection literature (e.g. [8, 26, 27]) is to make use of a detection specific overlap measure, such as the one used in the PASCAL VOC object detection challenge [7]. It has been noted that this overlap measure has several favorable properties compared to other measure such as invariance to scale and translation [13].

Desai et al. have taken an interesting approach in which the joint distribution between object detections is modeled linearly given features capturing statistics of the joint distribution of objects [6]. The model is trained discriminatively, but without approximation guarantees due to the greedy optimization employed in a cutting plane training algorithm. Their subproblem shares key characteristics with our random field characterization of non-maximal suppression, and the explicit characterization of a tractable family of models is a key contribution of the present work.

The approaches cited above largely work by employing sliding windows or other window subsampling strategies, but alternatively, variants on Hough transform detections have also been used. Leibe et al. proposed a widely adopted model in which visual words vote for an object center [24]. Gall and Lempitsky have developed a state of the art detection framework using Hough forests [9]. Lehmann et al. have recently presented a line of work that extends these models to efficient detection [22, 23] where the second citation uses branch-and-bound for optimization of detection. The present work in contrast is agnostic to the exact model employed, and the branch-and-bound framework we employ has been applied to several variants of non-linear models that cannot be represented using Hough transforms [20].

Barinova et al. have proposed a principled method of non-maximal suppression that can be interpreted as an explicit approximation to a full probabilistic model [1]. Their work is to our knowledge the first to couple approximation results for the maximization of submodular functions with object detection. Their work, however, is (i) restricted to models for which one can build a Hough image whereas the class of functions for which we can design a practical bound is more general, and (ii) their approach is restricted to very low dimensional detection parametrizations because Hough images are expensive to build for more than a few dimensions. Such an approach additionally must recompute a Hough image after each detection, while the proposed non-maximal suppression model can reuse the same data-structures (such as integral images [28, 20]) for subsequent detections.

Maximization of a submodular function with monotonic properties is common to many problems in computer science, from robotics [14] to social network analysis [16] and sensor networks [11, 18], and has been studied extensively in the operations research literature (a toolbox by Andreas Krause contains many of the algorithms developed there [17]). Branch and bound has been employed to find optimal solutions to the (in general) NP-hard problem [10], but has not, to our knowledge, been applied to greedy optimization of supermodular functions with optimal approximation guarantees, as in this work. The variety of problems that share the same structure promises that analogous optimization approaches to that proposed in this work may have wider application across computer science domains.

2 The Energy

We consider a very general class of joint energy functions that contains both an appearance model of the object class of interest, as well as terms incorporating beliefs about the joint distribution of object detections. These latter terms may be the result of a learning procedure, a prior over the joint positions of objects [6], or a set of constraints chosen *a priori* to disallow detections that have high overlap. We consider energies of the form

$$\max_y \sum_i \langle f, \phi(x, y_i) \rangle_{\mathcal{H}} - \mathbf{\Omega}(y). \tag{1}$$

Here we consider $\mathbf{\Omega}$ that factorizes into pairwise terms as well as higher order terms

$$\mathbf{\Omega}(y) = \sum_{ij} \Omega(y_i, y_j) + \underbrace{\sum_{c \in \mathcal{C}} \Omega_c(y_c)}_{\text{higher order terms}} \tag{2}$$

where x is an image, y_i is an object detection,[1] y is a collection of detections, ϕ is a joint kernel map, f is a function living in the RKHS defined by ϕ, Ω is a penalization term for detections that overlap too closely, and $c \in \mathcal{C}$ is a clique in the set of cliques contributing to the energy. In principle, higher order terms that are supermodular (see Section 3) do not affect the anaylsis in this paper. For simplicity, we will not treat them explicitly in the sequel.

We note that this form of energy for the detection of multiple objects may occur in diverse settings, such as object detection test time inference, detection cascades, and inference for cutting plane training of structured output learning [2, 15].

3 Minimization of a Supermodular Function

Many optimization approaches to random field models, such as graph cuts, rely on the submodularity of a function to be minimized. In the context of image segmentation, this is reflected in a general principle that neighboring pixels are likely to share the same label. Non-maximal suppression, however, enforces the exact opposite effect: neighboring detections are likely to have different labels, at least when the appearance term indicates an object is likely to be present in the vicinity.

In particular Equation (1) is the maximization of a submodular (minimization of a supermodular) function. Submodularity holds for a set function if for any two subsets of detections, A and B such that

$$A \subset B \tag{3}$$

the following holds

$$f(A \cup \{y\}) - f(A) \geq f(B \cup \{y\}) - f(B). \tag{4}$$

This is easy to show as

$$f(A \cup \{y\}) - f(A) = \langle f, \phi(x, y) \rangle_{\mathcal{H}} - \sum_{i \in A} \Omega(y_i, y) \tag{5}$$

$$\langle f, \phi(x, y) \rangle_{\mathcal{H}} - \sum_{i \in A} \Omega(y_i, y) \geq \langle f, \phi(x, y) \rangle_{\mathcal{H}} - \sum_{i \in B} \Omega(y_i, y) \tag{6}$$

$$0 \geq - \sum_{i \in B \setminus A} \Omega(y_i, y). \tag{7}$$

[1] In the sequel we pay particular attention to detections parametrized by bounding boxes.

Supermodular higher order terms in Equation (2) will be negated, resulting in submodularity. Equation (1) is therefore very difficult to optimize globally for multiple detections as maximizing a submodular (minimizing a supermodular) function is in general NP hard.

As our proposed optimization methodology is based on branch-and-bound, the practical constraints of its application to global optimization are key. Branch and bound ceases to be efficient due to curse of dimensionality for approximately 6 or more dimensions. While a bounding box provides a low (four) dimensional parametrization for single object detection, joint optimization of even two boxes leads to a combinatoric explosion of the complexity of the algorithm and is infeasible already for relatively small images. However, as has been exploited by Barinova et al. [1], strong theoretical results about the maximization of submodular functions indicates that a greedy approach gives optimal approximation guarantees for submodular energies [25]. Consequently, our optimization strategy will be to find the best detection without taking into account the non-maximal suppression terms, and then iteratively find subsequent detections, taking into account non-maximal suppression terms only with previously selected detections. The next section addresses the specific implications of this approach for branch and bound strategies, in particular how the structure of the problem can be exploited to improve the computational efficiency of subsequent detections.

4 Branch and Bound Implementations

Efficient subwindow search (ESS) is a branch and bound framework for object detection that works by storing sets of windows in a priority queue [19, 20]. Sets of windows are specified by intervals indicating the minimum and maximum coordinates of the four sides of the bounding box, and are ordered by an upper bound on the maximum score of any window within the set. This upper bound, $\hat{f}$, must satisfy two properties in order to guarantee the optimality of the result:

$$\hat{f}(Y) \geq f(y) \quad \forall y \in Y \tag{8}$$

$$\hat{f}(\{y\}) = f(y) \tag{9}$$

where Y is a set of bounding boxes specified by intervals for the sides of the box, and y is an individual window. The first property states that the upper bound is a true bound, while the second states that the score for a set containing exactly one window should be the true score of the window. Given these properties, when a state containing only one window is dequeued, we are guaranteed that this window has the maximal score of all windows in the image.

As we are pursuing a greedy optimization strategy, we wish to be able to compute upper bounds of the augmented quality function that contains both the unary terms, and the pairwise non-maximal suppression terms. Here, we discuss how to do so for a class of pairwise terms that are monotonic functions of the ratio of the areas of intersection and union of the two windows [7]

$$\Omega(y_i, y_j) = g\left(\frac{\text{Area}(y_i \cap y_j)}{\text{Area}(y_i \cup y_j)}\right) \tag{10}$$

where g is any non-negative monotonic function. Consequently, for the kth detection we require an upper bound for

$$\langle f, \phi(x, y_k) \rangle_{\mathcal{H}} - \sum_{i=1}^{k-1} \Omega(y_i, y_k) \tag{11}$$

where detections are ordered by their selection by the greedy optimization strategy. We may do so by taking the sum of two bounds, that of the unary terms, the construction of which is discussed for a number of linear and non-linear function classes in [20], and that of the non-maximal suppression term. The bound on the non-maximal suppression terms can be computed as

$$\max_{y \in Y} -g\left(\frac{\text{Area}(y_i \cap y)}{\text{Area}(y_i \cup y)}\right) \leq -g\left(\frac{\min_{y \in Y} \text{Area}(y_i \cap y)}{\max_{y \in Y} \text{Area}(y_i \cup y)}\right) \tag{12}$$

$$\leq -g\left(\frac{\min_{y \in Y} \text{Area}(y_i \cap y)}{(\max_{y \in Y} \text{Area}(y)) + \text{Area}(y_i) - (\min_{y \in Y} \text{Area}(y_i \cap y))}\right) \tag{13}$$

The computation of the bounds for area of overlap require only constant time given sets of windows specified by intervals.

A key property of greedy optimization of bounds of this form is that the objective for subsequent detections differs only by the subtraction of one additional Ω term. Since Ω is non-negative, this means that any valid bound for an earlier detection remains a valid upper bound for a subsequent detection (Equation (8)). This suggests that the computation required to find an earlier detection may be leveraged to more efficiently discover subsequent detections by *keeping the priority queue expanded by an earlier detection*. We also note, however, that Equation (9) may be violated if we simply continue the ESS branch-and-bound procedure without modification. This is because a state may be pushed into the priority queue containing only one window, but that does not consider non-maximal suppression terms resulting from detections discovered after that state was pushed into the queue. We can account for this by modifying the ESS algorithm in two ways: (i) we augment a state in the priority queue to store not only the upper bound and intervals specifying the set of bounding boxes, but also to store the number of previous detections considered in the computation of the upper bound, and (ii) we modify the termination criterion to check that the number of detections used for computation of the upper bound is equal to the number of detections found up to that point. If not, the bound is recalculated using all previous detections, and the state is re-inserted into the queue. We make a further assumption on the form of g for the purposes of subsequent analysis:

$$g(x) = \begin{cases} 0 & \text{if } x < \gamma \\ \infty & \text{otherwise} \end{cases} \tag{14}$$

where γ is a threshold on the overlap score (e.g. 0.5) above which multiple detections are disallowed. This results in the same non-maximal suppression criterion as used in recent state of the art detection strategies [8, 26, 27].

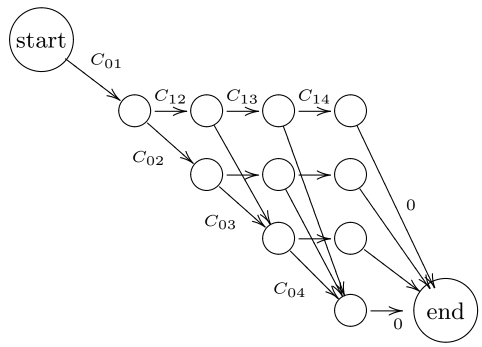


Fig. 1. Mapping of the selection of an optimal strategy to a shortest path problem. The resulting graph is constructed here for four detections. Horizontal moves correspond to keeping an existing priority queue for a subsequent detection, while diagonal moves correspond to resetting the priority queue to the root node containing the set of all bounding boxes. C_{ij} corresponds to the cost of computing the jth detection using the priority queue carried on from the ith detection. C_{0j} corresponds to the cost when resetting the priority queue prior to computing the jth detection. All edges pointing towards a given node have the same cost. This construction demonstrates that the complexity of computing the optimal strategy *given the branch-and-bound costs* are $\mathcal{O}(n^2)$ for n detections (see text). These costs are not known at test time, but we show empirically that optimal strategies have a very simple form (Section 6).

With these modifications, we can define a family of branch-and-bound strategies for multiple object detections. For each subsequent detection, a strategy may either reset the priority queue to contain a single state containing all possible windows in an image, or it may use a priority queue expanded from a previous detection (Figure 1). Each of these strategies will result in the *same set of detections*. Consequently, the goal is to determine a strategy or subset of strategies that reduces the expected computation time[2] of all detections. We fix the number of detections to 10 in this work and note that a strong pattern is apparent in the empirically observed computation times indicating that results are likely to generalize to other numbers of detections in real data.

5 Theoretical Results

Branch and bound can be characterized as a best-first search strategy over a DAG whose nodes are isomorphic to a Hasse diagram with direction assigned by set inclusion. We use the notation $\mathcal{Y}$ to indicate the maximal (root) element of the Hasse diagram containing all possible windows, Y to indicate a set of

[2] We use here the number of dequeuing operations required as a platform independent measure of the computation time. We note in particular that the bound computation is constant for the family of Ω considered here, making this a natural unit of measurement.

windows ($Y \subset \mathcal{Y}$, $|Y| > 1$), and y to indicate an individual window ($y \in \mathcal{Y}$). In practice, a subset of possible edges are considered corresponding to those such that Y can be represented by intervals. Furthermore, we consider a deterministic rule for splitting Y into two subsets following [19]. We denote the set of nodes visited by the best-first search from the root node with an upper-bound $\hat{f}$ as $S_{\hat{f}} \subset \mathcal{P}(\mathcal{Y})$, where $\mathcal{P}(\mathcal{Y})$ denotes the power set of $\mathcal{Y}$.

Theorem 1. *For valid upper bounds $\hat{f}_1$ and $\hat{f}_2$,*

$$\hat{f}_1(Y) \geq \hat{f}_2(Y) \quad \forall Y \implies S_{\hat{f}_2} \subseteq S_{\hat{f}_1} \tag{15}$$

Proof. Best first search expands all nodes with upper bound greater than the value of the true detection $f(y^*)$. $\hat{f}_2(Y) \geq f(y^*) \implies \hat{f}_1(Y) \geq f(y^*)$, but there may be additional Y for which $\hat{f}_2(Y) < f(y^*) \wedge \hat{f}_1(Y) \geq f(y^*)$.

Corollary 1. *$S_{\hat{f}_k} \subseteq S_{\hat{f}_i}$, where $k > i$ and $\hat{f}_k$ is a bounding function for the greedy optimization subproblem corresponding to detection k.*

Corollary 1 implies that there is a strict ordering of the number of nodes expanded by different objectives. As any priority queue expanded up to the point of an earlier detection will contain elements computed with a lose upper bound, we conclude that there is a potential computational advantage to resetting the priority queue to the root node for a subsequent detection. However, we also note that if the values of the function change only slightly, there will be a computational overhead to expanding the same nodes over again. Consequently, there may instead be a computational advantage to keeping an existing priority queue.

Stated simply, if we reset the queue to the root node we may have to re-expand nodes that had already been expanded in the previous round. If we don't reset the queue, we may have to go through a large number of nodes that have been expanded, but violate the non-maximal suppression condition in Equation (14).

Theorem 2. *The number of nodes to be re-expanded on reset of a queue for detection k is upper bounded by the sum of nodes expanded by other strategies up to that point.*

Proof. Nodes that have been previously expanded in round i can be categorized as belonging to one of two groups: (i) those for which $\hat{f}_i(Y) \geq f(y^*) \wedge \hat{f}_k(Y) \geq f(y^*)$ and (ii) those for which $\hat{f}_i(Y) \geq f(y^*) \wedge \hat{f}_k(Y) < f(y^*)$. All nodes in the first case will be expanded by both strategies, while nodes in the second case will be expanded by the previous detections, but not by the current detection.

The proof of Theorem 2 also indicates that in subsequent rounds after a reset, the marginal number of nodes to be expanded is strictly ordered, the older the priority queue, the more nodes will need to be expanded. This implies that once a priority queue has been reset and expanded until a subsequent detection is found, it will be superior to keep using that priority queue rather than one expanded from a previous set of detections.

These theoretical results indicate that for n detections, there are at most 2^{n-1} possible strategies of interest: for each detection after the first, we may either keep the existing priority queue with all expanded states, or we may reset the queue to the root node. If we were to know ahead of time all costs associated with a given choice, we could use a single-source shortest path algorithm to determine the optimal strategy. Figure 1 shows a mapping of the problem to a graph for four detections. As the graph is a DAG, the complexity of this procedure is $\mathcal{O}(V)$, where V is the number of vertices. For our graph construction, $V = \frac{n(n+1)}{2} + 2 = \mathcal{O}(n^2)$ resulting in an overall complexity of $\mathcal{O}(n^2)$ for n detections. This allows us *post hoc* to efficiently determine the optimal strategies in our empirical analysis.

This result unfortunately does not allow us to determine the lowest cost approach without precomputing all costs. Possible approaches would be to compute the empirical costs of these strategies for a sample of data, or to use a branch-and-bound strategy in the shortest path algorithm to avoid computing all edge costs. However, we show in Section 6 that *all* optimal strategies selected by this analysis on the PASCAL VOC data set have a simple form. This form consists of resetting the queue for a fixed number of initial detections, and then keeping the resulting priority queue without any resets for all subsequent detections. In practice, this indicates that only $n-1$ of the possible 2^{n-1} strategies are of interest.

6 Empirical Results

We present results for a modified implementation of the publicly available ESS code described in [20]. We use the feature extraction and trained models downloaded from the author's webpage. All results are reported on the test set of the PASCAL VOC 2007 data set [7], with a different objective trained for each of the 20 classes. Figure 2 shows the number of splits required for several selected classes, as well as the average across all classes for varying values of γ (Equation (14)). Figure 3 shows the number of splits conditioned on the presence or absence of the class of interest averaged across all classes. Table 1 shows statistics of the optimal strategy found by a shortest path search. For all classes, the optimal strategy consists of resetting the priority queue to the root node for a number of initial detections followed by re-using the existing priority queue for all subsequent detections. Table 2 shows the ratio of the amount of computation required by two simple strategies compared to the optimal strategy.

Table 1. Statistics of the number of resets to the root node required by optimal strategies. Statistics are reported across classes.

	$\gamma = 0.25$	$\gamma = 0.50$	$\gamma = 0.75$
min	3	2	1
median	4	3	2
max	4	4	3

Table 2. Ratios of the amount of computation required by two simple strategies to the optimal strategy. The first, naïve strategy consists of resetting the priority queue to the root node at each subsequent detection. The second strategy consists of keeping a single priority queue for all detections without any resets to the root node. Statistics are reported across classes.

$\gamma = 0.25$	**all reset**	**no reset**	$\gamma = 0.50$	**all reset**	**no reset**
min	1.36	1.17	min	1.38	1.16
median	1.48	1.22	median	1.52	1.20
max	1.94	1.28	max	2.19	1.28

$\gamma = 0.75$	**all reset**	**no reset**
min	1.59	1.14
median	2.04	1.16
max	3.15	1.20

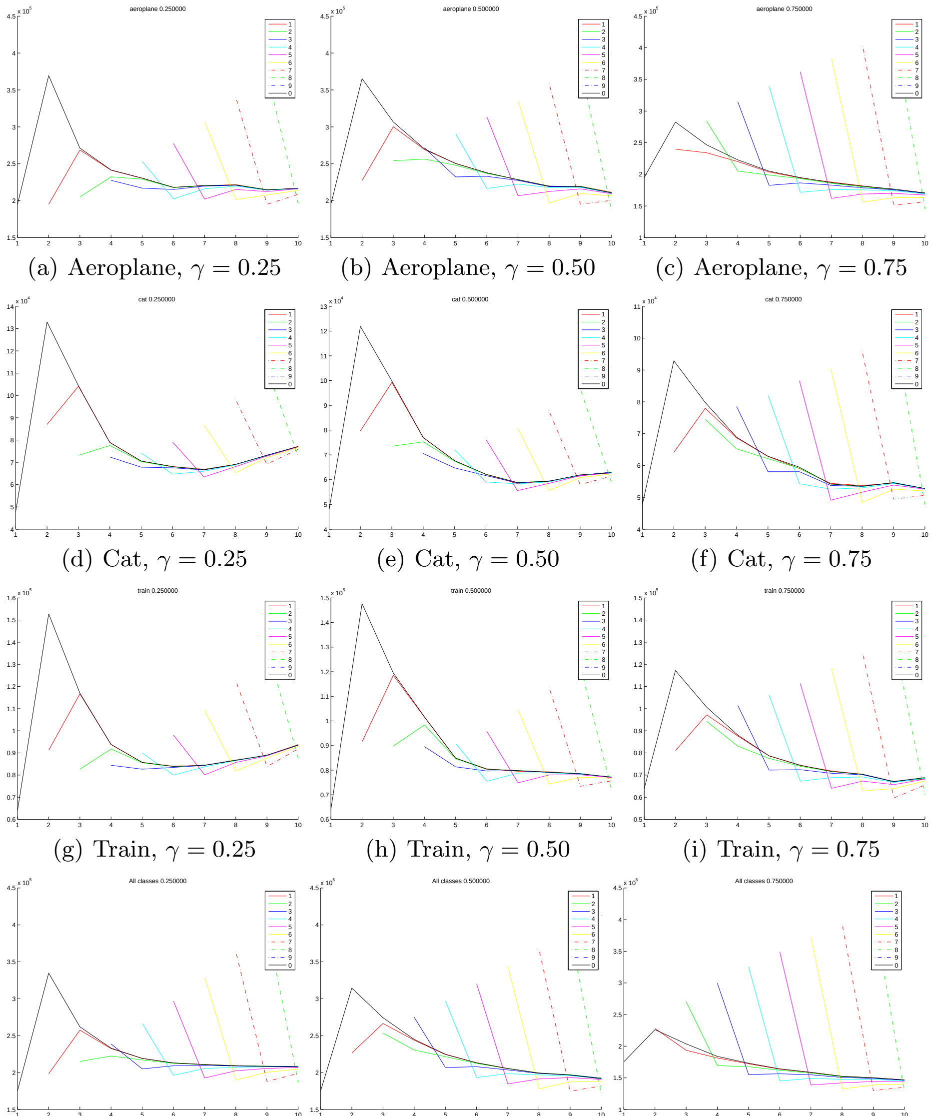
7 Discussion

Several broad conclusions can be drawn from the experiments reported in Section 6. The first, and most important for practical application of branch-and-bound to object detection with non-maximal suppression, is that there is a regime in which resetting the priority queue is more efficient than keeping an existing queue. However, after a few detections, ranging from one to four depending on the class of interest (Table 1), it is better to keep an existing priority queue for all subsequent detections. The proof of Theorem 2 indicates that more recently reset priority queues are *always* preferable to older queues. This has advantages, both in terms of the simplicity of the set of useful strategies, as well as in terms of reducing memory usage.

Varying behaviors were found when using differing values for γ. In general, the lower the value of γ (more strict non-maximal suppression) the more likely resetting the priority queue is beneficial. As γ increases from 0.25 to 0.75 the median number of resets taken by the optimal strategy for a given class decreases from 4 to 2. This makes intuitive sense as lower values of γ result in strictly higher numbers of nodes in the search graph that will be suppressed in subsequent branch-and-bound optimizations. A large number of expanded nodes around a peak will result in wasted computation as they are subsequently pruned by non-maximal suppression. Conversely, the higher the overlap threshold (less strict non-maximal suppression), the more likely keeping the existing priority queue is helpful.

Conditioning on the class label does not seem to show a large difference in the average number of splits per detection (Figure 3). This supports the idea that strategies may be fixed ahead of time.

The marginal cost of the first detection after resetting the priority queue to the root node is not strictly increasing (see e.g. Figure 2(d)), but is empirically observed to do so for many classes, and in the average performance across all



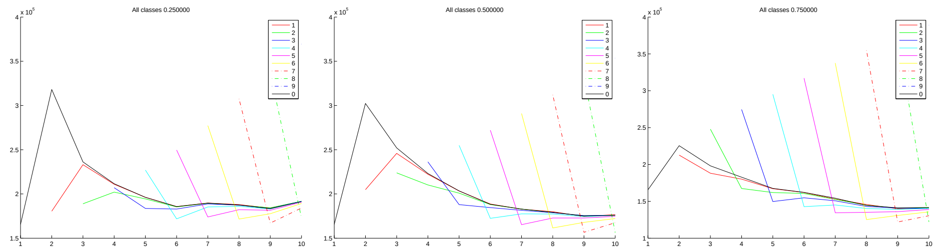
(a) Aeroplane, $\gamma = 0.25$ (b) Aeroplane, $\gamma = 0.50$ (c) Aeroplane, $\gamma = 0.75$

(d) Cat, $\gamma = 0.25$ (e) Cat, $\gamma = 0.50$ (f) Cat, $\gamma = 0.75$

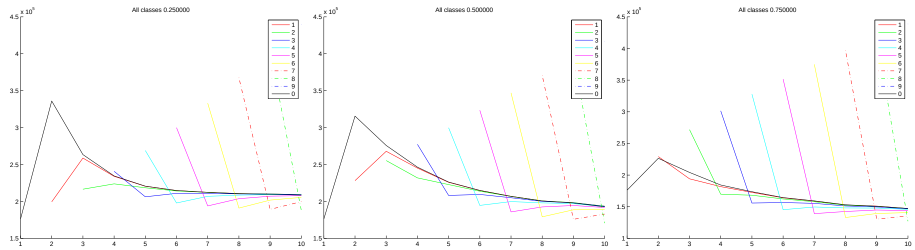
(g) Train, $\gamma = 0.25$ (h) Train, $\gamma = 0.50$ (i) Train, $\gamma = 0.75$

(j) All classes, $\gamma = 0.25$ (k) All classes, $\gamma = 0.50$ (l) All classes, $\gamma = 0.75$

Fig. 2. Number of splits per subsequent detection when resetting the priority queue at different detections vs. keeping an existing priority queue. x-axis: detection number, y-axis: average number of splits across all images in the VOC2007 test set.



(a) $+$, $\gamma = 0.25$ (b) $+$, $\gamma = 0.50$ (c) $+$, $\gamma = 0.75$



(d) $-$, $\gamma = 0.25$ (e) $-$, $\gamma = 0.50$ (f) $-$, $\gamma = 0.75$

Fig. 3. Number of splits per subsequent detection when resetting the priority queue at different detections vs. keeping an existing priority queue. x-axis: detection number, y-axis: average number of splits across all images and classes in the VOC2007 test set *conditioned on the presence or absence of an object of interest* (denoted $+$ and $-$, respectively).

classes (Figures 2(j)-2(l)). This result is in line with Theorem 2 which says that the upper bound on subsequent detections is increasing. This is especially apparent after the first few detections.

Finally, Table 2 indicates that of the simple strategies consisting of either always resetting the priority queue or never resetting the priority queue, it is preferable to never reset the priority queue. Our experiments showed that the amount of required computation for 10 detections was higher for each class and overlap threshold when using the resetting strategy than the simple strategy of always keeping the same priority queue.

8 Conclusions

Commonly applied non-maximal suppression strategies can be interpreted as optimization of a random field model in which non-maximal suppression is captured by pairwise terms encoding the joint distribution of object detection. We have shown in this work how to adapt a branch-and-bound strategy to optimize jointly over multiple detections with non-maximal suppression terms. An optimal approximation result allowed us to frame this as the subsequent application of inter-related branch-and-bound optimizations, enabling us to reuse computations across multiple detections. It is possible to frame the search for a computationally optimal strategy as a shortest path problem on a DAG with $\mathcal{O}(n^2)$ vertices, resulting in efficient *post hoc* computation of the optimal strategies. We have observed that these strategies have a very simple form: although every length $n-1$ bit string encodes a valid strategy resulting in 2^{n-1} possible strategies, all empirically optimal strategies consisted of first resetting the priority queue for a small number of detections, followed by keeping an existing priority queue. Furthermore, simply keeping a single priority queue for all detections resulted in only a modest increase in the total amount of required computation over the optimal strategy. This indicates that simple strategies can significantly improve computational performance over the naïve application of branch-and-bound in serial.

Acknowledgments

This work is supported by the Royal Academy of Engineering through a Newton International Fellowship.

References

1. Barinova, O., Lempitsky, V., Kohli, P.: On the detection of multiple object instances using Hough transforms. In: IEEE Conference on Computer Vision and Pattern Recognition (2010)
2. Blaschko, M.B., Lampert, C.H.: Learning to localize objects with structured output regression. In: Proc. ECCV (2008)
3. Blaschko, M.B., Vedaldi, A., Zisserman, A.: Simultaneous object detection and ranking with weak supervision. In: Proc. NIPS (2010)
4. Blaschko, M.B., Lampert, C.H.: Object localization with global and local context kernels. In: BMVC (2009)
5. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. CVPR (2005)
6. Desai, C., Ramanan, D., Fowlkes, C.: Discriminative models for multi-class layout. In: Proc. ICCV (2009)
7. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes (VOC) challenge. International Journal of Computer Vision 88(2), 303–338 (2010)
8. Felzenszwalb, P.F., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: Proc. CVPR (2008)
9. Gall, J., Lempitsky, V.: Class-specific hough forests for object detection. In: Proc. CVPR (2010)
10. Goldengorin, B., Sierksma, G., Tijssen, G.A., Tso, M.: The data-correcting algorithm for the minimization of supermodular functions. Management Science 45(11), 1539–1551 (1999)
11. Guestrin, C., Krause, A., Singh, A.: Near-optimal sensor placements in Gaussian processes. In: International Conference on Machine Learning (ICML) (August 2005)
12. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proc. of The Fourth Alvey Vision Conference. pp. 147–151 (1988)

13. Hemery, B., Laurent, H., Rosenberger, C.: Comparative study of metrics for evaluation of object localisation by bounding boxes. In: Image and Graphics, 2007. ICIG 2007. Fourth International Conference on. pp. 459 –464 (Aug 2007)
14. Hollinger, G., Singh, S.: Proofs and experiments in scalable, near-optimal search by multiple robots. In: Robotics: Science and Systems (June 2008)
15. Joachims, T., Finley, T., Yu, C.N.J.: Cutting-plane training of structural SVMs. Mach. Learn. 77(1) (2009)
16. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: Proc. KDD (2003)
17. Krause, A.: Sfo: A toolbox for submodular function optimization. Journal of Machine Learning Research 11, 1141–1144 (2010)
18. Krause, A., Guestrin, C., Gupta, A., Kleinberg, J.: Near-optimal sensor placements: Maximizing information while minimizing communication cost. In: International Symposium on Information Processing in Sensor Networks (IPSN) (April 2006)
19. Lampert, C.H., Blaschko, M.B., Hofmann, T.: Beyond sliding windows: Object localizationby efficient subwindow search. In: Proc. CVPR (2008)
20. Lampert, C.H., Blaschko, M.B., Hofmann, T.: Efficient subwindow search: A branch and bound framework for object localization. PAMI 31, 2129–2142 (2009)
21. Laptev, I.: Improvements of object detection using boosted histograms. In: Proc. ECCV (2006)
22. Lehmann, A., Leibe, B., van Gool, L.: Feature-centric efficient subwindow search. In: Proc. ICCV (2009)
23. Lehmann, A., Leibe, B., Van Gool, L.: Fast prism: Branch and bound hough transform for object class detection. International Journal of Computer Vision pp. 1–23 (2010)
24. Leibe, B., Leonardis, A., Schiele, B.: Combined object categorization and segmentation with implicit shape model. In: ECCV Workshop on Statistical Learning in Comp. Vision (2004)
25. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions. Mathematical Programming 14, 265–294 (1978)
26. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: Proc. ICCV (2009)
27. Vedaldi, A., Zisserman, A.: Structured output regression for detection with partial occulsion. In: Proc. NIPS (2009)
28. Viola, P., Jones, M.J.: Robust real-time object detection. IJCV (2002)