Bounding Part Scores for Rapid Detection with Deformable Part Models

Iasonas Kokkinos*

[1] Center for Visual Computing, École Centrale Paris, France
[2] Équipe Galen, INRIA Saclay, Île-de-France, France
[3] Université Paris-Est, LIGM (UMR CNRS), École des Ponts ParisTech, France

Abstract. Computing part scores is the main computational bottleneck in object detection with Deformable Part Models. In this work we introduce an efficient method to obtain bounds on part scores, which we then integrate with deformable model detection. As in [1] we rapidly approximate the inner product between a weight vector and HOG-based features by quantizing the HOG cells onto a codebook and replace their inner product with the lookup of a precomputed score. The novelty in our work consists in combining this lookup-based estimate with the codebook quantization error so as to construct probabilistic bounds to the exact inner product.

In particular we use Chebyshev's inequality to obtain probably correct bounds for the inner product at each image location. We integrate these bounds with both the Dual-Tree Branch-and-Bound work of [2,3] and the Cascade-DPMs of [4]; in both cases the bounds are used in a first phase to conservatively construct a short-list of locations, for which the exact inner products are subsequently evaluated.

We quantitatively evaluate our method and demonstrate that it allows for approximately a twofold speedup over both [2] and [4] with negligible loss in accuracy.

1 Introduction

Deformable Part Models (DPMs) [5,6,4] combine mathematical rigor with excellent performance, but come with increased computational cost when compared with simpler alternatives such as Bag-of-Word classifiers. In this work we accelerate detection with DPMs: we use the exact same 'mixture-of-DPMs' model as that of [7] and obtain a substantial speedup with negligible and controllable error.

Our work builds on [2,3] where 'Dual Tree Branch-and-Bound' (DTBB) is introduced as an efficient bounding-based method for detection with DPMs. DTBB substantially accelerates the stages following part computation for both single- and multi-category detection, turning their complexity from linear to roughly logarithmic in the image size. However, as highlighted in [8,4] and also mentioned in [2] the computation of part scores is actually the main computational bottleneck for DPMs.

In this work we introduce a method to rapidly compute upper and lower bounds for part scores. We integrate these part bounds in DTBB as well as in Cascade-DPM

* This work was supported by grant ANR-10-JCJC-0205.

A. Fusiello et al. (Eds.): ECCV 2012 Ws/Demos, Part III, LNCS 7585, pp. 41–50, 2012.

detection [4]. Our algorithm eventually computes the exact values of the part scores and the correct object score, but only around locations 'shortlisted' by a first bounding stage. This drastically reduces the number of exact score computations performed.

After presenting prior work in Sec. 2, we present our bounding technique in Sec. 3, and describe how we integrate our bounds with DTBB and cascaded detection in Sec. 4; in Sec. 5 we evaluate our method on the PASCAL VOC dataset. Our code is available at `http://vision.mas.ecp.fr/Personnel/iasonas/dpms.html`.

2 Prior Work on Efficient Object Detection

Cascade algorithms for detection were introduced in the beginning of the previous decade in the context of boosting [9] and coarse-to-fine detection in [10]. Cascades use a sequence of tests to decide about the presence of an object and stop whenever any of those test fails; this reduces the number of image locations where all tests are applied. Cascades use conservative thresholds, set to ensure that the number of false negatives on the training set (and hopefully also the test set) is minimized. This idea has been recently applied to detection with deformable models in [4,8] as well as to pose estimation with more complex structured models in [11], where thresholds are set in a learnable, data-dependent manner.

On the other hand, bounding-based techniques such as Efficient Subwindow Search (ESS) [12] bound the score of a detector within an interval by exploiting properties of its form; this allows to use techniques such as branch-and-bound, or coarse-to-fine search [13] to narrow-down the set of locations containing strong object hypotheses. Bounding-based (admissible) heuristics were used for hierarchical object parsing with A* in [14], while detection with deformable part models was cast in terms of Branch-and-Bound in [2,3]. More recently, [15] introduced a branch-and-bound method to deal with the more general structured models of [11].

The common idea in the works above is to prune computation when we have evidence that it is not worth pushing it until the end. For DPMs so far this has been done exclusively in 'cascade mode', namely by using empirically set thresholds. In the work of [4] on Cascaded Deformable Part Models (C-DPM) thresholds precomputed on the training set are used to prune computation with minimal possible loss. Specifically the authors observe that the DPM score is obtained gradually as the accumulation of the part scores; after computing the contribution of every part the authors stop the computation if the cumulative sum falls below a conservative (probably correct) threshold. In a similar vein [8] used a coarse-to-fine framework by using the 'root' (whole-object) filter response as a quick test to prune the range of locations where the part filters were evaluated. Again this requires setting a threshold for the root filter and fixing the range over which parts are searched for.

Using the results in our paper we can take an approach complementary to these two works: since we can bound the part scores we do not need to recompute thresholds. In particular for DTBB we approximate the part scores everywhere in a rapid manner and the optimization algorithm determines the subset of locations where the part score estimate is refined. For C-DPM we use our bounding-based technique to perform a rapid (faster than the one in [4]) computation of upper and lower bounds for the root

filter and then focus on a subset of part locations while also dynamically pruning the set of candidate object locations.

3 Part Score Bounding

3.1 Score Approximation

We compute part scores as inner products of Histogram-of-Gradient (HOG) [16] features with a part-specific weight vector, trained as in [7]. Denoting by $h[x, f]$ the f-th dimension of the HOG cell located at x, and by $w[y, f]$ the value of the 'part template' (weight vector) for location y and dimension f, the score of a part at x is:

$$s[x] = \sum_{y \in Y} \sum_{f=1}^{F} w[y, f] h[x + y, f], \tag{1}$$

where Y is a set of displacements and $F = 32$ is the dimensionality of the HOG cell at any point; we skip part indexes for simplicity. For 'part filters' $Y = [0, 5] \times [0, 5]$, so computing the score at any point x requires $36 \cdot 32$ multiplications and summations.

Our goal is to replace these $|Y| \cdot F$ operations with a rapidly computable approximation. By introducing the F-dimensional vectors $\mathbf{w}_y = [w[y, 1], \ldots, w[y, F]]^T$, and $\mathbf{h}_x = [\mathbf{h}[x, 1], \ldots, \mathbf{h}[x, F]]^T$, we can write the right hand side of Eq. 1 as:

$$s[x] = \sum_{y} \langle \mathbf{w}_y, \mathbf{h}_{x+y} \rangle. \tag{2}$$

As in [1], we use vector quantization to replace the F multiplications involved in every inner product with a single lookup operation that approximates the final outcome. For this we construct a codebook $\mathcal{C} = \{C_1, \ldots, C_K\}$ for $\mathbf{h}$ with K-means clustering (we use K =256) and create a $K \times |Y|$ array of precomputed values:

$$\mathit{\Pi}[k, y] = \langle C_k, \mathbf{w}_y \rangle, \tag{3}$$

which gives the score of part-cell y in the presence of an image-cell C_k.

When provided with a new image we quantize its HOG cells with our dictionary, i.e. we associate an index $I[x] = \mathrm{argmin}_k d(C_k, \mathbf{h}_x)$ with every image-cell $\mathbf{h}_x$; in particular we use KD-trees for fast approximate nearest neighbor search [17]. Given x we consider $\mathbf{h}_{i,j} \simeq C_{I[x]}$ and use use this approximation in Eq. 2 to obtain:

$$\langle \mathbf{h}_{x+y}, \mathbf{w}_y \rangle \simeq \langle C_{I[x+y]}, \mathbf{w}_y \rangle = \mathit{\Pi}[I[x + y], y] \tag{4}$$

$$s[x] \simeq \hat{s}[x] = \sum_{y} \mathit{\Pi}[I[x + y], y], \tag{5}$$

which exchanges the $|Y| \cdot F$ multiplications and summations of Eq. 2 with $|Y|$ lookup and summation operations. For $F = 32$ this can result in approximately a 30-fold speedup. In practice due to the numerous memory access operations the speedup can be smaller; our implementation is optimized to reduce the number of cache misses, but since the method is rather technical we will report it in a larger version of the paper; we refer to our publicly available code for details.

3.2 Bounding the Approximation Error

We now turn to ways of bounding the inner product score of Eq. 2 in terms of the lookup approximation in Eq. 5, by constructing upper and lower bounds on the approximation error. We first present a deterministic bound which comes from Holder's inequality. As this bound is too loose, it is only useful for presentation, and to introduce the necessary notation; we then provide tighter probabilistic bounds using Chebyshev's inequality.

3.3 A Deterministic Bound Using Holder's Inequality

We denote by $\mathbf{e}_x = \mathbf{h}_x - C_{I[x]}$ the quantization error for the feature $\mathbf{h}_x$ at x; We start by considering the contribution of a single HOG cell, say $x + y$ to the overall score; the approximation error ϵ_{x+y} in Eq. 4 can be expressed in terms of the inner product of $\mathbf{e}_{x+y}$ with $\mathbf{w}_y$ as follows:

$$\epsilon_{x+y} \doteq \langle \mathbf{h}_{x+y}, \mathbf{w}_y \rangle - \Pi[I[x+y], y] \tag{6}$$

$$= \langle \mathbf{h}_{x+y}, \mathbf{w}_y \rangle - \langle C_{I[x+y]}, \mathbf{w}_y \rangle = \langle \mathbf{e}_{x+y}, \mathbf{w}_y \rangle. \tag{7}$$

Holder's inequality states that $\|fg\|_1 \leq \|f\|_q \|g\|_p$ for $1/p + 1/q = 1$ and can be used to bound the approximation error in Eq. 7 in terms of the L_2-norms of the quantization error, $\|\mathbf{e}_{x+y}\|_2$ and the weight vector $\|\mathbf{w}_y\|_2$:

$$|\epsilon_{x+y}| = |\langle \mathbf{w}_y, \mathbf{e}_{x+y} \rangle| \leq \|\mathbf{w}_y\|_2 \|\mathbf{e}_{x+y}\|_2. \tag{8}$$

This result bounds the approximation error ϵ_{x+y} due to a single HOG cell. Coming to integrating the contributions form the multiple HOG cells, we turn to the approximation error in Eq. 4, which we denote as $\boldsymbol{\epsilon}_x \doteq s[x] - \hat{s}[x] = \sum_y \epsilon_{x+y}$ and bound as follows:

$$|\boldsymbol{\epsilon}_x| \leq \sum_y |\epsilon_{x+y}| \leq \sum_y \|\mathbf{w}_y\|_2 \|\mathbf{e}_{x+y}\|_2 \doteq B_x. \tag{9}$$

We note that computing B_x takes only $|Y|$ multiplications and summations: the norm of the quantization error is computed once for the whole image, independently of the number and size of parts, while the norms of the weight vectors can be precomputed.

3.4 A Probabilistic Bound Using Chebyshev's Inequality

Even though rapidly computable, the bound delivered above is too loose to be of practical use. This is due to its generality; as we now show, it can be tightened by exploiting our problem-specific knowledge. In particular the inner product in Eq. 7, $\langle \mathbf{e}_{x+y}, \mathbf{w}_y \rangle = \sum_f \mathbf{e}_{x+y}[f] \mathbf{w}_y[f]$ adds up the product of the weight vector and quantization error elements. We know that the distribution of $\mathbf{e}_{x+y}[f]$ will be zero-mean and consecutive summands are likely to cancel each other out. The Holder-based bound derived above ignores this and will be valid even if all vector elements have a common sign. As such it gives a bound which is too conservative to be useful.

Instead, we now derive a probabilistic bound that is valid with controllable accuracy. This means that we can tighten the bound at the cost of decreasing its probability of

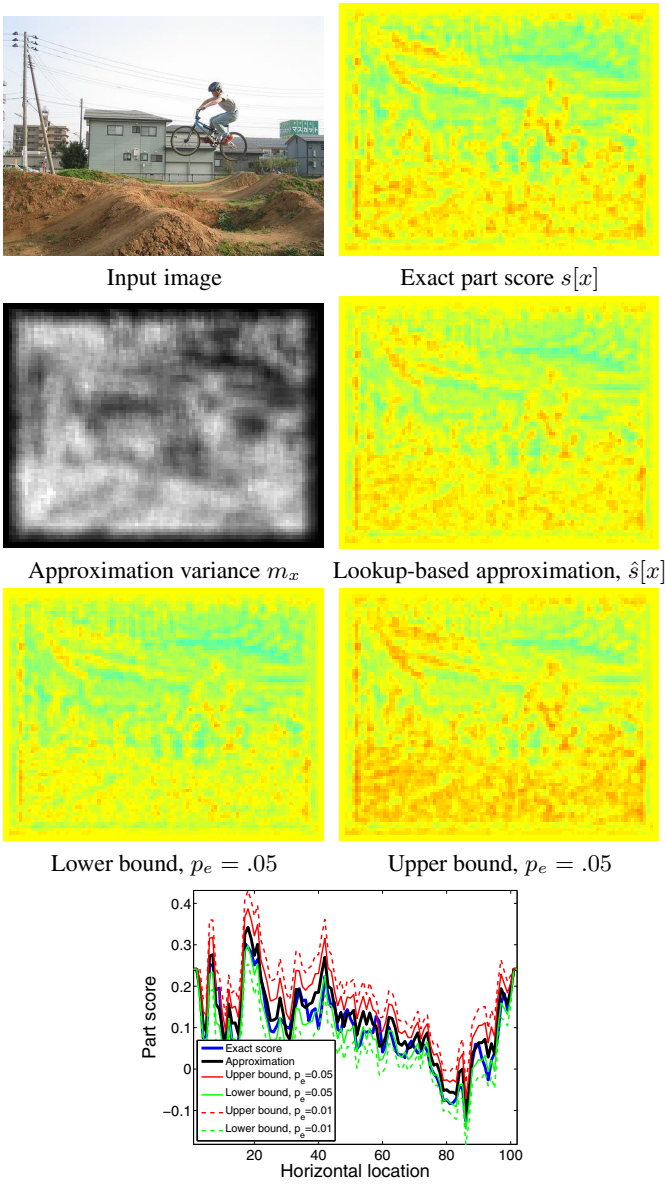

Fig. 1. Illustration of the part score approximation and bounding (please see in color): our goal is to rapidly bound the value of the part score $s[x]$ shown on the top right. The bound we propose in Eq. 12 is formed in terms of two quantities, a lookup-based approximation $\hat{s}[x]$ and an estimate of the approximation error's variance m_x, detailed in Eq. 13 and shown in the second row. These two are combined as in Eq. 12 to form an interval that contains the actual value with a certain probability of error p_e. The values of the lower and upper bounds for $p_e = 0.5$ are visualized in the third row. In the last row we show for an horizontal line through the image the values of the exact score, the approximation, as well as the upper and lower bounds for two different values of p_e. We can observe that for small p_e (higher probability of being correct) the bounds become looser.

being correct and vice versa. To obtain our bound we model the F elements of $\mathbf{e}_{x+y}$ and $\mathbf{w}_y$ involved in the inner product giving the HOG cell-level error:

$$\epsilon_{x+y} = \langle \mathbf{e}_{x+y}, \mathbf{w}_y \rangle \tag{10}$$

as samples from two distributions, $P_{x+y}(\mathbf{e}), P_y(\mathbf{w})$ respectively; since $\mathbf{e}$ is quantization error, $P_{x+y}(\mathbf{e})$ is considered to be zero-mean and symmetric around zero; we do not make assumptions about $P_y(\mathbf{w})$. We denote by $m^{\mathbf{e}}_{x+y}$ and $m^{\mathbf{w}}_y$ the respective variances (second moments) of the two distributions. Moreover, we assume that the quantization error at neighboring image locations is independent $\mathbf{e}_{x+y}$; one can consider cases where the quantization noise at neighboring locations has dependencies, e.g. when neighboring HOG cells are similar, and far outside the 'span' of the available codebook; a more thorough evaluation of whether this holds is needed, even though empirically we have observed that our subsequent bounds are valid.

Based on these assumptions, the products $\mathbf{e}_{x+y}[f]\mathbf{w}_y[f], f = 1, \ldots, F$ formed from the f-th elements of the vectors involved in Eq. 10 can be modeled as independent samples of a zero-mean, symmetric distribution with variance $m_{x+y} \doteq m^{\mathbf{e}}_{x+y} m^{\mathbf{w}}_y$. Consequently, the cell-level approximation error ϵ_{x+y} appearing in Eq. 10 can be seen as the sum of F independent variables having zero mean and variance m_{x+y}, so ϵ_{x+y} will in turn be a random variable with zero mean and variance Fm_{x+y}; similarly the part-level approximation error $\boldsymbol{\epsilon}_x$ will have zero mean and variance $m_x = \sum_y Fm_{x+y}$.

As per Chebyshev's inequality [18], any zero-mean random variable X satisfies:

$$P(|X| > \alpha) \leq \frac{E\{X^2\}}{\alpha^2}, \tag{11}$$

where $E\{\cdot\}$ denotes expectation - hence the numerator is the second moment of X. This means that with probability larger than $E\{X^2\}/\alpha^2$, X will be contained in $[-\alpha, \alpha]$; or, X will be contained in $[-\sqrt{E\{X^2\}/p_e}, \sqrt{E\{X^2\}/p_e}]$ with probability of error p_e.

We can use this fact to bound $\boldsymbol{\epsilon}_x$ probabilistically: with a probability of error p_e we will have $\boldsymbol{\epsilon}_x \in [-\sqrt{m_x/p_e}, \sqrt{m_x/p_e}]$. Since $\boldsymbol{\epsilon}_x = s[x] - \hat{s}[x]$, this means that with probability $1 - p_e$ we will have:

$$s[x] \in \left[\hat{s}[x] - \sqrt{m_x/p_e}, \hat{s}[x] + \sqrt{m_x/p_e}\right] \tag{12}$$

$$\text{where} \quad m_x = \sum_y F m^{\mathbf{e}}_{x+y} m^{\mathbf{w}}_y, \quad \hat{s}[x] = \sum_y \Pi[I[x+y], y]. \tag{13}$$

This bound is the main result of our paper. Comparing it to the Holder-based bound of Eq. 9, we first note that the empirical estimators of $m^{\mathbf{e}}_{x+y}, m^{\mathbf{w}}_y$ are related to the 2-norms of e_{x+y}, $\mathbf{w}_y$, respectively as:

$$m^{\mathbf{e}}_{x+y} = \frac{1}{F} \sum_{f=1}^{F} e^2_{x+y}[f] = \frac{1}{F} \|e_{x+y}\|_2^2 \tag{14}$$

and similarly $m^{\mathbf{w}}_{x+y} = \frac{1}{F}\|\mathbf{w}_{x+y}\|_2^2$. So apart from the root operation computing m_x in Eq. 13 has the same complexity as computing B in Eq. 9. Moreover the length of the interval in Eq. 12 scales proportionally to $\sqrt{|Y|F}$ while in Eq. 9 it scales proportionally to $|Y|F$, which shows that the Chebyshev bound is tighter than the Holder bound.

4 Integration with Deformable Object Detection

We now describe how we integrate the bound obtained above in the Dual-Tree Branch-and-Bound (DTBB) [2] and Cascaded DPM (C-DPM) [4] methods. Due to lack of space we refer to the respective works for algorithm details and provide a high-level, and intuitive outline of the integration, leaving technicalities for a future version.

4.1 Combination with Dual-Tree Branch-and-Bound

The DTBB method described in [2,3] uses bounding-based techniques for detection with DPMs. In particular Branch-and-Bound is used to bypass Generalized Distance Transforms (GDTs) and shown to result in substantial speedups for the part combination phase. However in [2] the part scores are considered to be computed in advance of DTBB, while this is actually the main computational challenge in detection with DPMs. Instead, we now integrate our efficient probabilistic part score bound with DTBB.

In particular, DTBB relies on upper bounding the quantity:

$$\mu_d^s = \max_{x \in X_d} \max_{x' \in X_s'} s[x'] + B[x', x] \tag{15}$$

which indicates the maximal contribution of a set of candidate part points X_s' to a set of candidate object points X_d; $s[x']$ is the appearance term at the candidate part location x' and $B[x', x]$ is the geometric consistency term between x' and the object location x. Since $\max_{x \in X} f[x] + \max_{x \in X} g[x] \geq \max_{x \in X} f[x] + g[x]$, we can bound Eq. 15 as:

$$\mu_d^s \leq \underbrace{\max_{x' \in X_s'} s[x']}_{\overline{S}} + \max_{x \in X_d} \max_{x' \in X_d} B[x', x] \tag{16}$$

A complementary term that emerges [3] is $\lambda_d^s = \min_{x \in X_d} \max_{x' \in X_s} s[x'] + B[x', x]$ which is lower bounded as:

$$\underbrace{\min_{x' \in X_s'} s[x']}_{\underline{S}} + \min_{x \in X_d} \max_{x' \in X_s} B[x', x] \leq \lambda_d^s. \tag{17}$$

The computation of the upper and lower bounds relevant to the geometric term, $B[x', x]$, exploits the fact that X_s, X_d are rectangular, and is detailed in [3]. Coming to the bounds on the unary terms, the approach taken has been to compute the exact part scores at every location $s[x]$ and then obtain $\overline{S}, \underline{S}$. As the domains X_s are organized in a kd-tree the latter maximization can be rapidly performed in a fine-to-coarse manner, but the computation of $s[x]$ was not avoided.

Instead we propose to accelerate the computation of $\overline{S}, \underline{S}$ by initially sidestepping the computation of $s[x]$ using the probabilistic bound of Eq. 13: the terms $\underline{s} = \hat{s}[x] - \sqrt{m_x/p}$ and $\overline{s} = \hat{s}[x] + \sqrt{m_x/p}$ involved in Eq. 13 are with probability $1 - p_e$ lower and upper bounds of $s[x]$ respectively. Based on these we can upper and lower bound S as follows: $\overline{S} = \max_{x' \in X_s'} s[x'] \leq \max_{x' \in X_s'} \overline{s}[x']$ and $\underline{S} = \min_{x' \in X_s'} s[x'] \geq \min_{x' \in X_s'} \underline{s}[x']$, and thereby use $\overline{s}, \underline{s}$ as surrogates for $s[x]$ in DTBB.

A subtlety is that by considering multiple terms when maximizing or minimizing with X_s we increase the probability of violating the (probabilistic) upper and lower bounds. But in practice we are only concerned with the points that give the maximum/minimum, and not the bulk of points contained between them.

We also note that we use $\overline{s}, \underline{s}$ as surrogates for s only in the first phase of DTBB. As soon as Branch-and-Bound converges to singleton intervals (individual pixels), we evaluate the exact part scores, $s[x]$; as we show in the experimental section, this boosts performance when compared to using only the lookup-based approximation. This more elaborate computation however is performed around a drastically reduced set of points, namely around those image locations that survive the first, quick bounding phase. Our method thereby combines the speed of quantization [1] with the accuracy of DTBB [2].

4.2 Combination with Cascaded Deformable Part Models

In [4] the authors exploit the fact that the DPM score is expressed as the accumulation of the part scores to devise a cascaded detection algorithm: after computing the contribution of each part to the overall object score, the computation stops for any location where the sum falls below a conservative (probably correct) threshold.

In order to accelerate the first stage of their algorithm, [4] downproject the HOG features to a lower dimensional space obtained through PCA. This results in fewer multiplications per HOG cell-bin, but can distort the obtained result. The remedy used in [4] is to use separate conservative thresholds for the PCA-based part scores, and estimate them from the training data.

By replacing the PCA-based approximation with the upper bound provided by our method we gain in two ways: first, our method is faster, as for each HOG-cell $x + y$ we use one lookup for $\Pi[I[x + y], y]$ and one multiplication for $\|\mathbf{w}_y\|_2 \|\mathbf{e}_{x+y}\|_2$ instead of 6 multiplications for the PCA-based features. Second, our method does not require gathering additional statistics to compute thresholds, but rather relies on the thresholds computed for the full (32-dimensional) part filters, which only need to be gathered once. This gives us the freedom to explore alternative bounding schemes (e.g. using different codebook construction techniques), without re-running our detector on the training data. Experimentally we verify that the two methods have virtually identical performance.

5 Results

We have validated the merit of our method on the PASCAL VOC'2007 challenge. As we use the exact same models as those in [4] our sole concern is the exactness and speed of the optimization method, and do not address learning issues.

In Fig. 2 we provide precision-recall plots for bicycle detection, in order to demonstrate the impact of our bounding scheme on object detection accuracy; similar results have been obtained for other classes but are omitted for lack of space; they will be included in a larger version of this work.

In all cases 'exact' refers to computations performed using GDTs as in [6]. On the left side, we compare the performance of the PCA-based cascade of [4] with the lookup-based cascade proposed in this work. We observe that if we use the 'raw' lookup-based

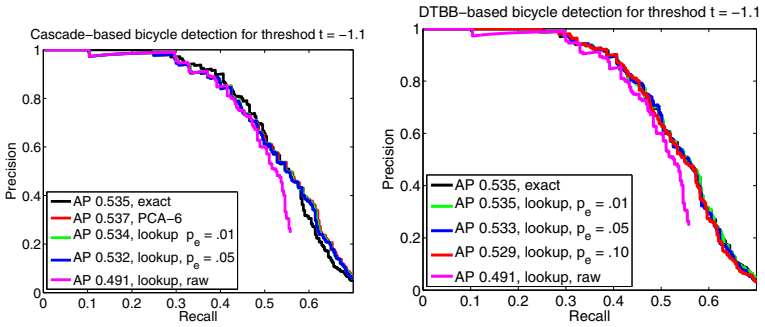

Fig. 2. Precision-Recall curves for bicycle detection using cascade-based (left) and branch-and-bound detection (right). Please see text for details.

Table 1. Means and standard deviation timings, in seconds, of the considered approaches. GDT stands for distance transforms, BB for Dual Tree Branch-and-Bound, CSC for cascade, and LU-{1,5} for lookup-based bounds with $p_e = .01$ and $p_e = .05$ respectively.

	GDTs [6]	BB [2]	BB-LU-5	BB-LU-1	CSC-PCA [4]	CSC-LU-5	CSC-LU-1
Part terms	8.35 ± 0.77	1.69 ± 0.18	0.69 ± 0.03	0.69 ± 0.06	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$\theta = -0.5$	0.60 ± 0.05	0.21 ± 0.06	0.47 ± 0.11	1.04 ± 0.25	0.56 ± 0.07	0.19 ± 0.03	0.23 ± 0.04
$\theta = -0.7$	0.60 ± 0.05	0.42 ± 0.10	1.00 ± 0.23	2.33 ± 0.65	0.72 ± 0.09	0.29 ± 0.04	0.36 ± 0.06
$\theta = -1.0$	0.60 ± 0.05	1.31 ± 0.31	3.80 ± 0.90	9.40 ± 2.70	1.04 ± 0.16	0.51 ± 0.10	1.07 ± 0.29

estimate of the part scores, without the related upper and lower bounds, performance drops significantly. However when using bounding intervals to accommodate the 'slack' due to the approximation error the performance directly becomes identical to the PCA-based cascade. However our method does not require additional threshold estimation, and as shown later is faster.

On the right plot we compare the performance of our lookup-based variant of DTBB for different values of p_e; we observe that for small values of p_e the performance is identical with GDTs, but with larger values of p_e performance decreases. Again, this validates the need for incorporating uncertainty in lookup-based approximations. This is consistent with the observations in [1] where performance was observed to drop, even when using a model directly trained with the lookup-based approximation to the features; it is all the more natural that performance drops when using a model trained with the full, clean features and testing with quantized features.

Coming to timing results, we provide in Table I timings gathered from 1000 images of the PASCAL VOC dataset, and averaged over all 20 categories. The first row indicates the time spent to compute part scores by the different methods, and the following rows indicate detection times. We observe that our lookup-based approximations are faster both for DTBB and Cascade Detection for moderate values of the threshold θ; in particular for $\theta = -.7$, or $\theta = -.5$ the lookup-based variant of cascades requires approximately half the time of the PCA-based cascade, and $1/30$ of the time of GDT-based detection. For more conservative threshold values the part score is fully evaluated at more points and the merits of the first fast pass get eliminated.

6 Conclusion

In this work we introduce Chebyshev's inequality to bound part scores in a simple and computationally efficient manner. We demonstrate the merit of our approach by combining the part score bounds with Branch-and-Bound and Cascade detection for deformable part models, which results in substantial speedups without loss in accuracy.

References

1. Vedaldi, A., Zisserman, A.: Sparse Kernel Approximations for Efficient Classification and Detection. In: CVPR (2012)
2. Kokkinos, I.: Rapid deformable object detection using dual-tree branch-and-bound. In: NIPS (2011)
3. Kokkinos, I.: Rapid Deformable Object Detection using Bounding-based Techniques. Technical Report 7940, INRIA (2012)
4. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A.: Cascade object detection with deformable part models. In: CVPR (2010)
5. Felzenszwalb, P.F., Huttenlocher, D.: Efficient Matching of Pictorial Structures. In: CVPR (2000)
6. Felzenszwalb, P.F., Girshick, R., McAllester, D., Ramanan, D.: Object Detection with Discriminatively Trained Part-Based Models. IEEE T. PAMI (2010)
7. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.: Discriminatively trained deformable part models, release 4, `http://people.cs.uchicago.edu/~pff/latent-release4/`
8. Pedersoli, M., Vedaldi, A., Gonzàlez, J.: A coarse-to-fine approach for fast deformable object detection. In: CVPR (2011)
9. Viola, P., Jones, M.: Rapid Object Detection using a Boosted Cascade of Simple Features. In: CVPR (2001)
10. Fleuret, F., Geman, D.: Coarse-to-fine face detection. IJCV (2001)
11. Sapp, B., Toshev, A., Taskar, B.: Cascaded Models for Articulated Pose Estimation. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 406–420. Springer, Heidelberg (2010)
12. Lampert, C., Blaschko, M., Hofmann, T.: Beyond sliding windows: Object localization by efficient subwindow search. In: CVPR (2008)
13. Lampert, C.H.: An efficient divide-and-conquer cascade for nonlinear object detection. In: CVPR (2010)
14. Kokkinos, I., Yuille, A.: HOP: Hierarchical Object Parsing. In: CVPR (2009)
15. Sun, M., Telaprolu, M., Lee, H., Savarese, S.: An efficient branch-and-bound algorithm for optimal human pose estimation. In: CVPR (2012)
16. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, vol. 2, pp. 886–893 (2005)
17. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms (2008), `http://www.vlfeat.org/`
18. Mitzenmacher, M., Upfal, E.: Probability and computing - randomized algorithms and probabilistic analysis. Cambridge University Press (2005)