The Fastest Deformable Part Model for Object Detection

Junjie Yan Zhen Lei Longyin Wen Stan Z. Li *
Center for Biometrics and Security Research & National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences, China
{jjyan,zlei,lywen,szli}@nlpr.ia.ac.cn

Abstract

This paper solves the speed bottleneck of deformable part model (DPM), while maintaining the accuracy in detection on challenging datasets. Three prohibitive steps in cascade version of DPM are accelerated, including 2D correlation between root filter and feature map, cascade part pruning and HOG feature extraction. For 2D correlation, the root filter is constrained to be low rank, so that 2D correlation can be calculated by more efficient linear combination of 1D correlations. A proximal gradient algorithm is adopted to progressively learn the low rank filter in a discriminative manner. For cascade part pruning, neighborhood aware cascade is proposed to capture the dependence in neighborhood regions for aggressive pruning. Instead of explicit computation of part scores, hypotheses can be pruned by scores of neighborhoods under the first order approximation. For HOG feature extraction, look-up tables are constructed to replace expensive calculations of orientation partition and magnitude with simpler matrix index operations. Extensive experiments show that (a) the proposed method is 4 times faster than the current fastest DPM method with similar accuracy on Pascal VOC, (b) the proposed method achieves state-of-the-art accuracy on pedestrian and face detection task with frame-rate speed.

1. Introduction

The deformable part model (DPM) [11] is one of the most popular object detection methods. It is originally proposed for Pascal VOC [9] challenge and is the foundation of champion systems in Pascal VOC 2007-2011. Recent works have extended DPM to related tasks and achieved leading performance, such as articulated human pose estimation [35], face detection [36, 34] and pedestrian detection [33, 32]. DPM has advantage in handling large appearance variations for challenging datasets, however, it takes more than 10 seconds (without parallelization) per image in Pascal VOC. The speed is a bottleneck of DPM in real application, where speed is often as important as accuracy.

Recent works have accelerated deformable part model (DPM) by one order of magnitude, such as cascade [10], coarse-to-fine [22], branch-and-bound [16] and FFT [8]. In DPM, the detection score of each hypothesis is determined by the score of appearance minus the deformation cost. The appearance score is calculated by the correlation between HOG feature and a sequence of filters including root and parts, which takes most of the time due to the high dimension. [10] and [22] reduced the computation by pruning unpromising hypothesis early. [8] used FFT to accelerate correlation. These methods, however, still take about 1 second per image for Pascal VOC detection. We take cascade DPM [10] as the baseline, and find that each step in 2D correlation by root filter, cascade part pruning and HOG feature extraction makes DPM prohibitive even if the other two steps are free. To finally remedy the speed bottleneck of DPM, we thus need accelerate all these three steps.

Discriminative Low Rank Root Filter In DPM (and accelerated versions [10, 22, 16]), root scores are densely computed by 2D correlation between the root filter and HOG feature map. This paper reduces the cost by constraining the rank of root filter. As used in other computer vision tasks [13, 21, 31, 25], the 2D correlation can be divided into linear combination of more efficient 1D correlations, where the combination number is the rank of the filter. To learn the low rank filter while preserving the discriminative ability, an additional nuclear norm is added to traditional SVM objective function. This paper adopts a proximal gradient algorithm to progressively learn it by minimizing an upper bound function with closed form solution. The discriminatively learned low rank root filter can reduce the correlation cost and help to prune a large number of negative hypotheses.

Neighborhood Aware Cascade DPM can be more efficient through cascade based pruning of low score hypotheses after evaluating a subset of parts, as explored in [10]. However, there are still two kinds of redundancy in this cascade. The first is that one object can activate multiple

*corresponding author

overlapping hypotheses to pass through the whole cascade, while only one hypothesis with the highest score is useful for detection. The second is that if one hypothesis has very low score, its neighborhoods tend to have low scores and probably can avoid evaluation. Motivated by the crosstalk [5] in boosting classifier, this paper proposes neighborhood aware cascade for DPM to reduce the two kinds of redundancy. Many hypotheses in this cascade can be aggressively pruned according to the first order approximation of stage scores by their neighborhoods, instead of explicit computation.

Look-up Table HOG HOG is used in DPM as a low-level representation due to the advantage in tolerating local transformation. However, the original HOG calculation has high computational cost, mainly due to the operations in calculating the orientation partition and magnitude. This paper shows that look-up table (LUT) can be used to replace them with much simpler matrix index operations, based on the fact that there are only finite possibilities of gradient and orientation.

The rest of the paper is organized as follows. Section 2 reviews the related work. An overall introduction of DPM is presented in section 3. The discriminative low rank root filter, neighborhood aware cascade and LUT HOG are described in section 4, 5, 6 respectively. We show experiments in section 7 and conclude the paper in section 8.

2. Related Work

Acceleration of DPM This work is most related to approaches that accelerate single category DPM in detection. [10] proposed to convert star-structure to cascade, which can efficiently prune unpromising hypotheses. [22] proposed a coarse-to-fine approach based on that model at low resolution can prune a lot of hypotheses with low computational cost. FFT was used to accelerate the correlation in [8]. Motivated by the branch-and-bound approach [20] for object detection, [16] introduced it to DPM with carefully designed bound. For a category with 6 components, these methods run at about 1 FPS per Pascal VOC image on a single thread, which is faster than DPM by one order, but still relatively slow for real application.

Acceleration of Multi-category DPM Quite a large number of recent works [23, 27, 15, 3, 17] were proposed to accelerate DPM for multi-category detection, e.g. simultaneous detection of 20 categories on Pascal VOC. Steerable part model [23] used a part bank with linear combination to approximate correlation score of different categories. Sparselet [27, 15] used a large part bank with sparse linear combination. [15] and [23] both achieve three times acceleration over the original DPM for 20 category object detection, however, they are slower than the cascade DPM [10] which detects each category independently. Very recently, [3] proposed to use locality-sensitive hashing to approximate the correlation in DPM with a decline of performance to detect 100,000 categories on a single workstation.

Acceleration of Pedestrian Detection Recently, large improvements on efficiency were achieved in pedestrian detection task [6, 5]. [6] proposed to approximate features at nearby scales for fast computation of multiple channel features. Based on the feature and boosting classifier in [6], [5] further proposed crosstalk cascade by considering the dependence in neighborhood. [5] is considered to be one of the best detectors in Viola-Jones framework [30] in terms of speed and accuracy. We extend this idea to neighborhood aware cascade in DPM.

HOG computation The widely used HOG implementation in [12] takes about 0.5 second per VGA image on a single thread, which itself slows down DPM. Unfortunately, this step is often directly ignored by recent works on acceleration of deformable part model. Some recent works [27, 28, 24] accelerated HOG with the computation capacity of GPU, however, the algorithm itself is not improved. With the help of LUT, HOG implementation in this paper runs on a single CPU thread is as fast as the GPU implementation reported in [24]. LUT based method can be applied on GPU for more acceleration.

3. DPM and Cascade DPM

This part gives a brief review of DPM and cascade DPM, and then analyzes the bottleneck in computation.

The DPM is composed of a root filter w_0 and n parts, where the t-th part is parameterized by filter w_t and deformation term d_t. An object hypothesis γ is specified by $\{p_0, p_1, \cdots, p_n\}$, where p_0 is the location of root, and p_t is the location of the t-th part. Root and parts are connected by a pictorial structure. The detection score $s(\gamma)$ is defined as:

$$s(\gamma) = w_0^T \phi_a(p_0, I) + \sum_{t=1}^{n} w_t^T \phi_a(p_t, I) - d_t^T \phi_d(p_t, p_0), \tag{1}$$

where ϕ_a is the HOG feature for appearance, and ϕ_d is separable quadratic function for deformation. Mixture components can be naturally added to represent objects in different poses, but we leave them out to simplify the notation.

For a hypothesis γ in detection, only root location p_0 is known, while the part location p_t is inferred by maximizing the part appearance score minus the deformation cost associated with displacement:

$$p_t = \arg\max_p w_t^T \phi_a(p, I) - d_t^T \phi_d(p, p_0), \tag{2}$$

where p traverses possible locations of the part. Since parts are directly attached to the root, their locations are inferred independently for a fixed root. It has been found in previous works [10, 22, 8] that in DPM most of the time is spent on calculating the appearance term due to the high dimension.

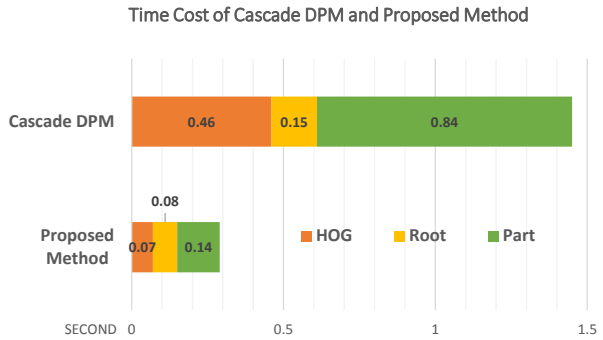

Figure 1. Average time cost (second) of the cascade DPM [10] and the proposed method on Pascal VOC with a single CPU thread. The time for HOG extraction, root and part are listed, while other steps are nearly free. For each category, the DPM has 6 mixture components and each component has 8 parts.

In cascade DPM [10], acceleration is achieved by reducing the number of parts evaluated. The cascade DPM places the root filter in the first stage and part filters sequently in following stages. In each stage, hypothesis can be pruned if its score is below a pre-learned threshold. The time cost of each step in cascade DPM is shown in Fig. 1. It has accelerated DPM by one order, however, all these three steps are still prohibitive.

In the first stage of cascade, correlation is calculated densely between root filter and feature map, while in following part stages, correlation is only calculated sparsely for unpruned hypotheses. In this paper, different methods are used to accelerate the two kinds of computation. We learn discriminative low rank root filter in first stage for both efficient dense correlation and safe pruning of unpromising hypotheses after the correlation. For parts, we design more aggressive pruning by exploring the first order neighborhood information. Besides, the HOG feature extraction is also accelerated by look-up tables. With these three acceleration techniques, the proposed method is 4 times faster than the cascade DPM (shown in Fig. 1). We show details of these three techniques in following three sections.

4. Discriminative Low Rank Root Filter

In this part, we aim to reduce the cost on computation of root score, which is the result of dense correlation between HOG feature map and root filter. The acceleration comes from the separability and linearity of correlation. Suppose we have a 2D feature map $K \in R^{m_1 \times n_1}$ and a 2D filter $F \in R^{m_2 \times n_2}$ with rank r ($r \leq \min(m_2, n_2)$). With the SVD decomposition, F can be expressed as:

$$F = \sum_{i=1}^{r} \sigma_i u_i v_i^T, \tag{3}$$

where $\sigma_i \in R$ is the i-th singular value. Herein $u_i \in R^{m_2 \times 1}$, $v_i \in R^{n_2 \times 1}$. With this expression, it can be easily proved that:

$$K \circ F = K \circ \sum_{i=1}^{r} \sigma_i u_i v_i^T = \sum_{i=1}^{r} \sigma_i ((F \circ u_i) \circ v_i^T), \tag{4}$$

where $\circ$ denotes the correlation operator. In the last term, the correlation is firstly conducted on each column by 1D filter u_i, and then on each row by 1D filter v_i. This procedure requires $r(m_2 + n_2)m_1 n_1$ multiplications[1], while the original 2D correlation need $m_2 n_2 m_1 n_1$ multiplications. It is easy to see that the combination of two 1D correlations in Eq. 4 needs fewer multiplications if the rank r is small enough, so that we expect to have low rank root filter for computation efficiency.

Besides the low rank property, the ability to distinguish objects and backgrounds is also preferred for the root filter, in order to efficiently prune negative hypotheses after the correlation. In the following part, we describe how to learn this kind of discriminative low rank root filter.

Matrix based representations are used to simplify the notation. Let the dimension of HOG cell be l. The i-th 2D HOG feature plane of root specified by γ of image I is denoted as $\{\Phi_i(\gamma, I)\}_{1 \leq i \leq l}$, and the i-th 2D plane of root filter is denoted as $\{W_i\}_{1 \leq i \leq l}$. We denote $diag\{W_i\}_{1 \leq i \leq l}$ as W and $diag\{\Phi_i(\gamma, I)\}_{1 \leq i \leq l}$ as $\Phi(\gamma, I)$. The correlation score can then be denoted as $Tr(W^T \Phi(\gamma, I))$, where $Tr(\cdot)$ is the trace operator. One traditional way to get the discriminative root filter is SVM based learning, where the filter is expected to distinguish true object hypotheses from backgrounds. Given M training samples, W can be learned by SVM:

$$\min_W \frac{1}{2}\|W\|_F^2 + C \sum_M max(0, 1 - y_m Tr(W^T \Phi(\gamma_m, I_m))), \tag{5}$$

where $\|\cdot\|_F$ is the Frobenius norm. $y_m \in \{-1, 1\}$ is the label of hypothesis specified by γ_m and I_m, where 1 indicates object and -1 indicates background. The first term is used for regularization, and the last term is used to measure the loss in detection.

As aforementioned, the root filter is desired to be low rank for efficient correlation computation. Motivated by recent works on matrix completion [2], we use additional nuclear norm to constrain the rank of W in learning:

$$\begin{aligned} &\min_W \mu\|W\|_* + \frac{1}{2}\|W\|_F^2 \\ &+ C \sum_M max(0, 1 - y_m Tr(W^T \Phi(\gamma_m, I_m))), \end{aligned} \tag{6}$$

where $\|\cdot\|_*$ is the nuclear norm. μ controls the trade-off between the efficiency and loss in detection. Despite Eq. 6 being convex, there are two difficulties in optimization. The first is the large number of negative samples, for which we use the hard sample mining procedure similar to [11]. The

[1] In implementation, u_i is replaced with the "dot product" result $\sigma_i \cdot u_i$ to avoid additional multiplications of σ_i at runtime.

second is the convergence when training set is fixed, for which we adopt the proximal gradient method [29].

Denoting the sum of the last two terms in Eq. 6 as $f(W)$ (which is also convex), one sub-gradient of $f(W)$ can be formulated as:

$$\nabla f(W) = W + C\sum_{M} h(W, \gamma_m, y_m, I_m), \tag{7}$$

where $h(W, \gamma_m, y_m, I_m)$ is set to be:

$$\begin{cases} 0 & if \quad y_m Tr(W^T \Phi(\gamma_m, I_m)) \geq 1 \\ -y_m \Phi(\gamma_m, I_m) & otherwise. \end{cases} \tag{8}$$

Defining Y as a local region of W, the sub-gradient satisfies the Lipschitz condition $\|\nabla f(W) - \nabla f(Y)\|_F \leq L_f \|W - Y\|_F$, where a conservative L_f is set to be cM (which is a constant in this problem), since that $\|\Phi(\gamma_m, I_m)\|_F$ can be naturally bounded by a constant c. A quadratic approximation of the objective function in Eq. 6 by Taylor expansion can be formulated as:

$$\begin{aligned} Q(W, Y) &= \mu\|W\|_* + f(Y) \\ &+ Tr(\nabla f(Y)^T (W - Y)) + \frac{L_f}{2}\|W - Y\|_F^2. \end{aligned} \tag{9}$$

It can be easily proved that $Q(W, Y)$ is the tight upper bound of the Eq.6 due to the Lipschitz condition of sub-gradient. Defining a matrix $G = Y - \frac{1}{L_f}\nabla f(Y)$, Eq. 9 can be minimized by the following problem instead:

$$\arg\min_W Q(W, G) = \arg\min_W \mu\|W\|_* + \frac{L_f}{2}\|W - G\|_F^2. \tag{10}$$

Suppose the SVD decomposition of G is $U\Sigma V^T$, the closed form solution to Eq. 10 can be obtained as (see [1]):

$$W = U D_\tau(\Sigma) V^T, \tag{11}$$

where $D_\tau(\Sigma) = diag(\{max(\sigma_i - \tau, 0)\})$, and $\tau = \mu / L_f$. The Eq. 9 can be iteratively optimized according to a sequences of $\{Y_k\}$ once the the training set is fixed. In each iteration, following the advice in [29], we set $Y_k = W_k + \frac{t_{k-1}-1}{t_k}(W_k - W_{k-1})$, and $t_k = \frac{1+\sqrt{4t_{k-1}^2+1}}{2}$.

The details of the optimization procedure in root filter learning are shown in Alg. 1. We use standard hard negative sample mining procedure according to the learned W in the outer loop, and then refine W by mined samples in the inner loop. The initialization of W is set to be root filter used in the first stage of cascade DPM [10], which is the original DPM root filter with PCA dimension reduction. In experiments, the rank of filter in each plane is 2 or 3, which is about 1 times faster than the original full rank filter for correlation. Moreover, since that the root filter is discriminatively learned, it is able to prune many negative hypotheses. Note that the root filter at the first stage of cascade is not necessarily to be optimal for the whole DPM, and we can re-compute scores for unpruned hypotheses with original root filter at later stage.

Algorithm 1 Proximal Gradient Algorithm for Discriminative Low Rank Root Filter Learning

Input: We set W_0 and W_1 to be the root filter after PCA in original DPM. $t_0 = t_1 = 1$ and $k = 1$. Initial training set $\{\Gamma_M, I_M\}$ is initialized by all positive samples and sampled negative samples.

Output: Discriminative low rank root filter W

while Not Converged **do**
  Mine hard negative samples with W to update the training set $\{\Gamma_M, I_M\}$.
  while Not Converged **do**
    $Y_k \leftarrow W_k + \frac{t_{k-1}-1}{t_k}(W_k - W_{k-1})$.
    $\nabla f(Y_k) \leftarrow Y_k + C\sum_m h(W_k, \gamma_m, y_m, I_m)$
    $G_k \leftarrow Y_k - \frac{1}{L_f}\nabla f(Y_k)$.
    $(U_k, \Sigma_k, V_k) \leftarrow svd(G_k)$
    $W_{k+1} \leftarrow U_k D_{\frac{u}{L_f}}(\Sigma_k) V_k^T$
    $t_{k+1} \leftarrow \frac{1+\sqrt{4t_k^2+1}}{2}$, $k \leftarrow k+1$
  end while
  $W \leftarrow W_t$.
end while

5. Neighborhood Aware Cascade

In this part, we focus on the reduction of parts computation cost by neighborhood aware cascade.

Cascade DPM [10] improves the efficiency by pruning unpromising hypotheses early. Starting from the calculation of root score $s_0(\gamma)$ in the first stage for each hypothesis γ, parts are evaluated sequently in following stages. The score of the t-th ($t \geq 1$) stage is defined as:

$$s_t(\gamma) = s_{t-1}(\gamma) + w_t^T \phi_a(p_t, I) - d_t^T \phi_d(p_t, p_0), \tag{12}$$

where each stage evaluates a part. There are two pruning criteria in [10]. The hypothesis γ can be pruned directly if the t-th stage satisfies that $s_t(\gamma) < \rho_t$, where ρ_t is a predefined threshold. By traversing optimal part location in a local region, the deformation pruning is adopted if the score $s_t(\gamma)$ minus deformation cost is below ζ_t. However, there are still two kinds of redundancy can be reduced for further acceleration, as discussed in the following part.

The first redundancy, which is always ignored by previous works, exists in evaluating positive hypotheses. It is well known that an object instance always active multiple overlapping detections. A merge step such as non-maximal suppression (NMS) is usually adopted to eliminate these overlapping hypotheses and finally preserve the detection with the highest score. The redundancy is that we only need one per overlapping detection group, but all of them pass the whole cascade. In experiments we test the cascade DPM and find that one final detection corresponds to average 21.85 detections before NMS step (with default threshold). We name these eliminated overlapping positive hypotheses as semi-positive hypotheses. They can take about half of the

time (most hypotheses in later stages belong to this case), and we want to prune them early to save computation.

The second redundancy exists in evaluating negative hypotheses. In traditional cascade based pruning, each hypothesis is evaluated independently. Nevertheless, the procedure ignores the fact that there has great dependency among detection scores in neighborhood regions. For example, a hypothesis with very low score indicates that its neighborhoods probably have very low score and do not need to be evaluated any more. We name these negative hypotheses with low score neighborhood as semi-negative hypotheses and want to prune them before explicitly evaluating their scores at certain stages.

Motivated by [5], we use the "first order" information in DPM cascade pruning to avoid the two kinds of redundancy. That is, besides explicitly calculating stage score, we can also estimate it according to their neighborhoods by first order approximation. We name this cascade as neighborhood aware cascade. Let the neighborhood of γ be $N(\gamma)$. We add the following two first order pruning criteria to decide whether γ is pruned or passed to next stage (the formal proofs can be found in supplementary material).

Semi-Positive Pruning: If $\exists\gamma' \in N(\gamma)$ which satisfies that $s_t(\gamma') > s_t(\gamma)+\mu_t$, γ is pruned without evaluating left stages. Herein μ_t is a pre-learned threshold. It is reasonable since that if score of a hypothesis is much lower than its neighborhood, it will be pruned in NMS step even it passed all the cascade.

Semi-Negative Pruning: If score of a hypothesis at the t-th stage is below a threshold $s_t(\gamma) < \nu_t$, all the hypotheses in its neighborhood region $N(\gamma)$ are pruned without evaluating. This is because the score of a hypothesis can be bounded by its neighborhoods under first order approximation.

The details of the neighborhood aware cascade for DPM are listed in Alg. 2. $Z(\gamma)$ in Alg. 2 indicates whether γ is pruned or not. The neighborhood $N(\gamma)$ is set to be a 5×5 region centered at γ empirically after cross-validation. The algorithm is started from root score computation with the learned low rank root filter. The lines 9-15 in Alg. 2 are used to find the best part location by searching a local region $\Delta(p_0, t)$ and add its score. In the final step, we also use NMS to merge overlapping hypotheses, but the number is much fewer than the cascade DPM. In implementation, similar to [10], PCA is used to simplify the appearance in early stages, and then original full filters are used at late stages. Another useful detail is that part scores can be cached to avoid repeated calculation by its neighborhoods.

To learn the thresholds $\{\mu_t, \nu_t, \rho_t, \zeta_t\}$, we run original DPM detector on labeled object hypotheses and their neighborhoods, and cache their scores of root and parts. The optimal threshold should be as large as possible for aggressive pruning, but must ensure not prune true object hypotheses.

Algorithm 2 Neighborhood Aware Cascade in DPM

Input: Pre-learned thresholds $\{\mu_t, \nu_t, \rho_t, \zeta_t\}$, hypothesis set Γ of an input image I, index set Z with all value initialized by 1.
Output: Detection set D

1: Calculate the root score of all hypotheses in first stage by dense correlation between feature map and low rank root filter.
2: **for** $t = 1$ to n **do**
3: **for** $\gamma \in \Gamma$ & $Z(\gamma) = 1$ **do**
4: **if** $s(\gamma) \leq \nu_t$ **then**
5: $Z(N(\gamma)) \leftarrow 0$
6: **else if** $s(\gamma) \leq \rho_t$ or $s(N(\gamma)) - s(\gamma) > \mu_t$ **then**
7: $Z(\gamma) \leftarrow 0$
8: **else**
9: $f \leftarrow -\infty$
10: **for** $p \in \Delta(p_0, t)$ **do**
11: **if** $s(\gamma) - d_t^T\phi_d(p, p_0) > \zeta_t$ **then**
12: $f \leftarrow max(f, w_t^T\phi_a(p, I) - d_t^T\phi_d(p, p_0))$
13: **end if**
14: **end for**
15: $s(\gamma) \leftarrow s(\gamma) + f$
16: **end if**
17: **end for**
18: **end for**
19: $D \leftarrow \text{NMS}(\Gamma(Z = 1))$

Let the object hypothesis training set be X, we set $\rho_t = \min_{\gamma\in X} s_t(\gamma)$, and $\zeta_t = \min_{\gamma\in X}(s_t(\gamma) - d_t^T\phi_d(p_t, p_0))$, where $d_t^T\phi_d(p_t, p_0)$ is the deformation cost. The μ_t and ν_t are defined based on neighborhoods of labeled positive hypotheses. We set $\mu_t = \min_{\gamma\in X}(s_t(\gamma) - \max(s_t(N(\gamma))))$ and $\nu_t = \min_{\gamma\in X} s_t(N(\gamma))$. Although it is better to learn thresholds from a new validation set, we find that learning thresholds from the training set is good enough in experiments.

We note that in experiments, the semi-negative pruning mainly appears in early stages, the semi-positive pruning mainly appears in later stages, and the traditional "zero order" pruning appears in all stages. A comparison between cascade [10] and proposed neighborhood aware cascade on the number of pruned parts in each stage is shown in Fig. 2. We also try to use neighborhood aware pruning in the first stage for root (instead of dense correlation in line 1 of Alg. 2), but we find that it is not as efficient as dense low rank correlation.

6. LUT HOG

In this part we show how to dramatically reduce the computation cost while generating exactly the same HOG feature.

The HOG feature map is constructed on each scale independently by resizing input image. For each scale, the pixel-wise feature map, spatial aggregation and normalization are operated in sequence. In pixel-wise feature map

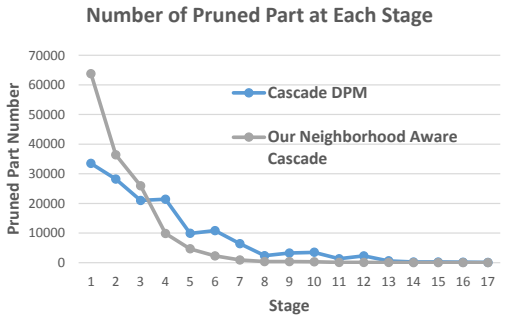

Figure 2. Average pruned part number at each stage on VOC 2007.

step, the gradient of each pixel is discretized into different partitions according to the orientation. In spatial aggregation step, the gradient magnitude of each pixel is added to its corresponding bins in four cells around it with bilinear interpolation weight. Finally, a normalization step is applied to gain the invariance by normalizing feature vector with energy of four cells around. By analyzing a popular and well optimized implementation in [12], we find that the first two steps takes most of the time. The analysis here is also valid for implementations in [8, 22].

We use look-up table (LUT) to accelerate the first two steps for HOG. With the LUT, the runtime computation is replaced with simpler and more efficient array indexing operation. It is based on the fact that the pixels in image are represented by "uint8" integral numbers. They can only generate limited cases of gradient orientation and magnitude, so that can be computed in advance and stored as part of model initialization. LUT is also valid for the computation of the bilinear interpolation weight in spatial aggression step since that the possible bilinear weight number is the HOG bin size.

Take the pixel-level feature map computation for example. Since pixels are in range of $[0, 255]$, the gradients at x and y directions are in range of 511 integers $[-255, 255]$. We pre-calculate three 511×511 look-up tables T_1, T_2 and T_3, where T_1, T_2 and T_3 store the index of contrast sensitive and insensitive orientation partition, and the magnitude for possible gradient combinations in x and y directions, respectively. In runtime, these three values for each pixel can be indexed in T_1, T_2 and T_3 instead of explicit computation.

The LUT based HOG computation is very simple and easy for implementation. Our implementation based on LUT is 6 times faster than the implementation in [11] on the same hardware, which clears up the time bottleneck in computing HOG feature.

7. Experiments

To evaluate the speed and accuracy of the proposed method, experiments are conducted on Pascal VOC 2007 object detection task [9]. Due to the special interests on pedestrian and face in real applications, we also conduct experiments on challenging Caltech pedestrian detection task [7] and AFW face detection task [36].

7.1. Pascal VOC 2007

On Pascal VOC 2007, the proposed method is implemented based on DPM release4[2] [12]. Besides the implementation of DPM release4, we compare accelerated DPM versions, including cascade [10], branch-bound [16], coarse-to-fine [22] and FFT [8]. All these methods except coarse-to-fine [22] use the default setting and model in DPM release4, where the number of levels in an octave is 10, HOG bin size is 8, part number for each component is 8 and component number for each category is 6. For coarse-to-fine DPM, the setting advised by the paper [22] is used, where component number is 4. The average feature extraction time, detection time and full time of the 20 categories are reported in Tab. 1, where the detection time sums the root and parts computation time. For fair comparison, all the codes run on the same PC with 2.66GHz Intel X5650 CPU, and only one thread is used in reporting Tab. 1. The accuracy on Pascal VOC 2007 testset (shown in Tab. 2) is measured by average-precision (AP) [9].

Table 1. Average time (measured by second) on Pascal VOC 2007. Note that 6 components are used for each category and the times are measured on a single thread implementation.

	Feature Extraction	Detection	Full Time
DPM [12]	0.46	11.77	12.23
Branch-Bound (DPM) [16]	0.46	2.75	3.21
Cascade (DPM) [10]	0.46	0.99 (0.15+0.84)	1.45
FFT (DPM) [8]	0.48	0.98	1.46
Coarse-to-fine (DPM) [22]	0.67	0.99	1.66
Proposed Method	0.07	0.22 (0.08+0.14)	0.29

Different DPM methods get similar accuracy on Pascal VOC. Cascade [10], FFT [8] and coarse-to-fine [22] get similar 10 times acceleration over the DPM release4. With three acceleration techniques proposed in this paper, the proposed method runs 4 times faster than these accelerated DPM methods. Compared with the cascade DPM, proposed method takes 1/2 time in calculation of root, 1/6 time in calculation parts and 1/7 time in calculation of HOG feature. Proposed method runs at 3-4 FPS for a category with 6 components per image on Pascal VOC. When parallelization is allowed, e.g. one thread for a component, the speed of proposed method is up to 15 FPS.

One may also be interested in the comparison between Viola-Jones based detector and proposed method for object detection. Detectors are trained on Pascal VOC based on the state-of-the-art Viola-Jones style detector ACF [4], with DPM style mixture components. Although ACF is one

[2]We use release4 instead of release5, mainly due to that most algorithms compared are based on release4. Generally speaking, release5 would give a slight higher accuracy with exactly the same speed.

Table 2. Average-Precision (AP) of different methods on 20 categories of Pascal VOC 2007 testset.

	plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	mean
DPM [12]	29.2	56.1	9.9	16.5	24.6	45.7	54.9	17.2	21.6	23.1	14.4	10.3	57.6	47.6	41.9	12.3	18.0	28.2	44.2	40.1	30.7
Branch-Bound (DPM) [16]	24.1	56.1	0.0	9.1	22.2	42.1	53.6	9.1	19.2	16.2	9.1	9.1	56.7	46.0	40.0	9.1	9.1	24.5	42.3	37.2	26.7
Cascade (DPM) [10]	27.6	56.2	9.9	16.6	24.7	45.5	55.0	17.3	21.6	22.8	14.4	10.4	57.7	48.0	41.8	12.3	18.1	28.6	44.3	40.1	30.6
FFT (DPM) [8]	30.1	56.2	9.8	15.0	23.7	48.3	54.8	16.4	22	22.4	18.1	10.5	56.3	46.4	40.9	12.4	17.7	29.7	42.6	37.2	30.5
Coarse-to-fine (DPM) [22]	27.9	54.8	10.2	16.1	16.2	49.7	48.3	17.5	17.2	26.4	21.4	11.4	55.7	42.2	30.7	11.4	20.9	29.1	41.5	30.0	28.9
Proposed Method	27.1	57.9	9.9	16.1	24.2	45.2	54.1	17.1	20.9	22.7	14.4	10.3	57.1	47.8	41.5	12.2	18.1	27.8	44.2	38.5	30.4

times faster (i.e., 0.12s per image) than proposed method, it can only get half the accuracy (i.e., 15.4 mean AP).

7.2. Caltech Pedestrians

Caltech pedestrian benchmark [7] is one of the most challenging pedestrian detection task due to large appearance variations in occlusion, pose, deformation and resolution. It is taken as a testbed to compare proposed method with other state-of-the-art methods for pedestrian detection. Following the protocol in [7], set00-set05 are used to train model and set06-10 are used for test. The "reasonable" setting in [7] is used to report the performance, where pedestrians above 50 pixels in height of each 30 frames are taken into consideration.

We report ROC and mean miss rate of the top methods[3] plus Viola-Jones and HOG in Fig. 3. Since this paper just considers the frame-wise detection, only methods without usage of in-frame and between-frame context are compared. The proposed method is on par with the best performance method MT-DPM [33] and outperforms the Viola-Jones style detector ACF [4] by 2%. These three methods largely outperform other methods. We compare the speed of these top three methods. The number of scales evaluated per octave is 5 and the mixture component number is 1, which are good enough for pedestrian detection task. In this setting, the proposed method runs at 10 FPS, while the MT-DPM runs at 1.2 FPS with FFT based acceleration. The well optimized ACF runs at 21 FPS with lower accuracy. When 6 cores are used for parallelization (mainly for HOG feature in this experiment), speed of the proposed method is about 40 FPS, which is fast enough for most applications.

7.3. AFW Faces

The proposed method is also validated on AFW face detection task [36]. It contains 205 images with 468 faces in the wild. Model in proposed method is trained on AFLW dataset [18]. Training faces are split into 6 components based on the pose annotations provided in [18] with yaw angles in [0°,30°), [30°,60°), [60°,90°] and their mirrors. Similar to the configuration for Pascal VOC, 8 parts are used for each component.

Recall-precision curve and average precision are used to report the performance. The results from [36] and a very recent work [26] are used for comparison. Note that the TSM (tree structure model [36]) and DPM reported in [36] are trained on Multi-PIE, while the proposed method is trained by more wild faces from AFLW. As shown in Fig. 4, the proposed method obtains a 93.7% AP on AFW, which is better than Face.com and very close to Google Picasa. The proposed method is about 100 times faster than TSM [36]. Although accuracy is not the main concern of the paper, the proposed method is better than TSM [36] by 5% AP. For full yaw pose face detection in VGA image, proposed method runs at 5 FPS on a single thread and 25 FPS if 6 threads are used. If only frontal faces are concerned, proposed method runs about 11 FPS (single thread) or 42 FPS (after parallelization), which approximates the speed of Viola-Jones detector in OpenCV [4]. Considering the large performance gain and similar speed, the proposed method has the potential to replace Viola-Jones detector for face detection in the wild.

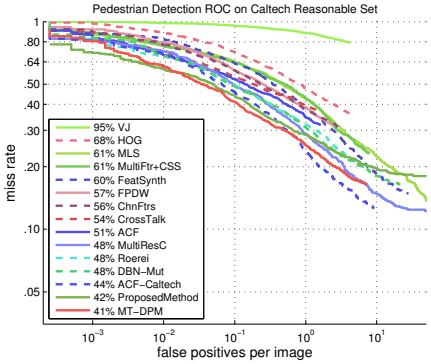

Figure 3. ROC curve and mean miss rate of leading methods on Caltech "reasonable" testset. We only report pure detection methods (without context) for fair comparison. (Best viewed in color)

8. Conclusion

In this paper, three novel techniques are proposed to solve the speed bottleneck of deformable part model, while maintaining its advantage in accuracy for various detection tasks. The proposed method runs at 4 times faster than the previous fastest DPM method on Pascal VOC. For pedestrian and face detection, it runs at frame-rate with state-of-

[3]Details can be found in Dollár's website http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/.

[4]We note that Google Picasa has similar time cost when running the software.

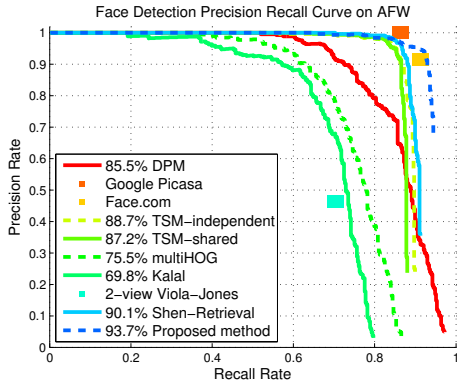

Figure 4. Face detection Precision-recall and average precision on AFW dataset. The proposed method dramatically outperforms the methods reported in [36] and Face.com, and very close to Google Picasa. (Best viewed in color)

the-art accuracy, i.e. 10 FPS on a single CPU thread and 40 FPS after parallelization. We expect this work can extend the DPM to real applications, such as video surveillance and HCI. Techniques discussed in this paper can also be used to accelerate related models, such as deep convolutional network [19, 14], which is taken as one of the future work.

Acknowledgement

We thank the anonymous reviewers for their valuable feedbacks. This work was supported by the Chinese National Natural Science Foundation Projects 61105023, 61103156, 61105037, 61203267, 61375037, National Science and Technology Support Program Project 2013BAK02B01, Chinese Academy of Sciences Project KGZD-EW-102-2 and AuthenMetric Research and Development Funds.

References

[1] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 2010. 4

[2] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 2011. 3

[3] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013. 2

[4] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *PAMI*, 2014. 6, 7

[5] P. Dollár, R. Appel, and W. Kienzle. Crosstalk cascades for frame-rate pedestrian detection. In *ECCV*. Springer, 2012. 2, 5

[6] P. Dollár, S. Belongie, and P. Perona. The fastest pedestrian detector in the west. *BMVC 2010*, 2010. 2

[7] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *PAMI*, 34, 2012. 6, 7

[8] C. Dubout and F. Fleuret. Exact acceleration of linear object detectors. In *ECCV*. Springer, 2012. 1, 2, 6, 7

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 1, 6

[10] P. Felzenszwalb, R. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *CVPR*. IEEE, 2010. 1, 2, 3, 4, 5, 6, 7

[11] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 2010. 1, 3, 6

[12] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. http://people.cs.uchicago.edu/ pff/latent-release4/. 2, 6, 7

[13] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *TPAMI*, 1991. 1

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 8

[15] R. B. Girshick, H. O. Song, and T. Darrell. Discriminatively activated sparselets. In *ICML*, 2013. 2

[16] I. Kokkinos. Rapid deformable object detection using dual-tree branch-and-bound. In *NIPS*, 2011. 1, 2, 6, 7

[17] I. Kokkinos. Shufflets: Shared mid-level parts for fast object detection. In *ICCV*. IEEE, 2013. 2

[18] M. Kostinger, P. Wohlhart, P. Roth, and H. Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *ICCV Workshops*. IEEE, 2011. 7

[19] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 8

[20] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *PAMI*, 2009. 2

[21] R. Manduchi, P. Perona, and D. Shy. Efficient deformable filter banks. *TSP*, 1998. 1

[22] M. Pedersoli, A. Vedaldi, and J. Gonzalez. A coarse-to-fine approach for fast deformable object detection. In *CVPR*. IEEE, 2011. 1, 2, 6, 7

[23] H. Pirsiavash and D. Ramanan. Steerable part models. In *CVPR*. IEEE, 2012. 2

[24] V. Prisacariu and I. Reid. fasthog - a real-time gpu implementation of hog. Technical report, Oxford University, 2009. 2

[25] R. Rigamonti, V. Lepetit, and P. Fua. Learning separable filters. In *CVPR*. IEEE, 2012. 1

[26] X. Shen, Z. Lin, J. Brandt, and W. Ying. Detecting and aligning faces by image retrieval. In *CVPR*. IEEE, 2013. 7

[27] H. O. Song, S. Zickler, T. Althoff, R. Girshick, M. Fritz, C. Geyer, P. Felzenszwalb, and T. Darrell. Sparselet models for efficient multiclass object detection. In *ECCV*. Springer, 2012. 2

[28] P. Sudowe and B. Leibe. Efficient Use of Geometric Constraints for Sliding-Window Object Detection in Video. In *ICVS'11*, 2011. 2

[29] K.-C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 2010. 4

[30] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 2004. 2

[31] L. Wolf, H. Jhuang, and T. Hazan. Modeling appearances with low-rank svm. In *CVPR*. IEEE, 2007. 1

[32] J. Yan, Z. Lei, D. Yi, and S. Z. Li. Multi-pedestrian detection in crowded scenes: A global view. In *CVPR*. IEEE, 2012. 1

[33] J. Yan, X. Zhang, Z. Lei, S. Liao, and S. Z. Li. Robust multi-resolution pedestrian detection in traffic scenes. In *CVPR*. IEEE, 2013. 1, 7

[34] J. Yan, X. Zhang, Z. Lei, D. Yi, and S. Z. Li. Structural models for face detection. In *FG*. IEEE, 2013. 1

[35] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*. IEEE, 2011. 1

[36] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*. IEEE, 2012. 1, 6, 7, 8