Efficient Discriminative Learning of Parts-based Models

M. Pawan Kumar*
Computer Science Dept.
Stanford University
pawan@cs.stanford.edu

Andrew Zisserman
Dept. of Engineering Science
University of Oxford
az@robots.ox.ac.uk

Philip H.S. Torr
Dept. of Computing
Oxford Brookes University
philiptorr@brookes.ac.uk

Abstract

Supervised learning of a parts-based model can be formulated as an optimization problem with a large (exponential in the number of parts) set of constraints. We show how this seemingly difficult problem can be solved by (i) reducing it to an equivalent convex problem with a small, polynomial number of constraints (taking advantage of the fact that the model is tree-structured and the potentials have a special form); and (ii) obtaining the globally optimal model using an efficient dual decomposition strategy. Each component of the dual decomposition is solved by a modified version of the highly optimized SVM-Light algorithm. To demonstrate the effectiveness of our approach, we learn human upper body models using two challenging, publicly available datasets. Our model accounts for the articulation of humans as well as the occlusion of parts. We compare our method with a baseline iterative strategy as well as a state of the art algorithm and show significant efficiency improvements.

1. Introduction

Parts-based models have been a popular choice for representing object categories, at least since the seminal work of Fischler and Eshclager [12]. They have been adapted for several fundamental problems of computer vision such as object recognition [10], object detection [5] and pose estimation [7]. The two main challenges that face researchers who employ these models are (i) how to learn them efficiently using training data; and (ii) how to fit them to test images. Several advances in the optimization literature have now enabled us to address the latter problem of fitting parts-based models [7, 8]. However, the problem of learning the models still remains open.

The approaches adopted so far can broadly be divided into two categories: generative and discriminative learning. Generative models are typically learnt by maximizing the likelihood of positive training data (i.e. images containing an instance of the object category) [5, 7, 10, 22]. They offer the advantage of allowing the user to generate samples of the model which may prove useful when the goal is to marginalize the model, e.g. see [16]. In contrast, discriminative learning attempts to learn a model that can distinguish between positive and negative training data (i.e. images that do not contain the object). In practice discriminative models provide accurate results [9, 28], often outperforming generative models in similar settings. Inspired by this observation, we develop a supervised, discriminative approach for efficiently learning the model using large amounts of training data. Specifically, we learn the weighting of the shape, appearance and spatial configuration of the model such that it maximizes the margin between the pose of the object in the positive images (provided by the user during training) and all possible poses in the negative images.

The main problem to be addressed here is that of handling the large number of possible poses of the object in the negative images. We show how this seemingly difficult task can be reduced to that of solving a series of small convex optimization problems by (i) considering tree-structured models; and (ii) taking advantage of the form of spatial configuration generally used in such model, e.g. Potts, (truncated) linear, or (truncated) quadratic. These restrictions might not allow the model to fully capture the statistics of the object category. However, as observed in previous work [7, 9, 10], the advantages of these models (efficient learning and testing) outweigh their disadvantages.

Related work. Several methods have been proposed for learning discriminative models. They range from approaches that provide a local minimum of their corresponding optimization problem (for example, by discarding spatial information [1] or by treating each part separately [19]) to globally optimal algorithms for convex optimization problems. For instance, Ramanan and Sminchisescu [21] describe a gradient ascent method for maximizing the conditional likelihood of the training data. However, their approach does not explicitly maximize the margin between positive and negative data. The breakthrough work on max-margin Markov networks $M^N$ [24] formulates the problem of maximizing the margin as a convex optimization problem. Recently, the $M^N$ formulation has also been extended to mixtures of probabilistic models [30]. Given the training data, $M^N$ is learnt using an iterative algorithm where each iteration chooses a subset of variables to be optimized. However, choosing the right subset of variables involves solving an inference problem\footnote{Here we use inference to specifically refer to the process of fitting the current model to each training or test image.} at each iteration. Furthermore, the overall number of variables in the optimization problem grows quadratically with the number of possible poses of the parts. This makes the approach of [24] inefficient for our current model to each training or test image.
task where the number of poses is large.

In terms of the problem formulation, our method is most related to the supervised case discussed in [9]. As observed in [9], this problem is convex, albeit with the number of constraints that are exponential in the number of parts. In order to overcome this difficulty [9] proposes an iterative strategy which starts with a small number of constraints. At each iteration it adds a subset of constraints that were most violated by the solution of the previous iteration. This is similar in spirit to the strategy adopted for solving structured output support vector machines (SVM) [27] which have recently been successfully used for learning whole-object models [3], parts-based models [26] and for low-level vision [17, 23]. However, the identification of the most violated constraints again involves solving an inference problem on each training image at each iteration.

**Novelty.** The above methods should only be employed when inference can be performed efficiently (e.g. in [9] where the corresponding potentials can be computed by simple convolutions of the given image with the filter of each part, or in [23] where dynamic graph cuts are employed). In contrast, our approach provides a more attractive option in cases where performing inference is computationally expensive, e.g. in the case of articulated object categories where several rotations of each part need to be considered while computing the potentials, or when occlusion is explicitly modelled (since it effectively doubles the number of putative poses of a part, see section 4). This is due to the fact that we avoid performing inference by considering all the constraints of the optimization problem of [9] simultaneously. Although the number of constraints is exponential in the number of parts, we show that for tree-structured models with Potts pairwise potentials they can be reduced to a small, polynomial number of constraints. We obtain the globally optimal model by solving the resulting optimization problem using dual decomposition [2]. Each component of the dual decomposition is similar to the SVM learning problem [29] and hence, can be solved efficiently.

We use our method to learn human upper body models with two challenging, publicly available datasets. However, the methodology presented here is equally applicable to any other object category (e.g. facial features, quadrupeds, and rigid/semi-rigid objects such as bicycles and airplanes). We provide a comparison with a baseline iterative strategy [18] and the method used in [9] and show significant improvements in both learning from training images and pose estimation in test images.

### 2. Supervised Learning

Given a set of labelled training images, we consider the problem of learning a discriminative parts-based model. Here, by labelled we mean that each training image has been labelled as either positive (i.e. containing an instance of the object category of interest) or negative (i.e. background image). Furthermore, in the case of positive images, we are also provided with the poses of the parts. The pose of a part is represented as \((x, y, \phi)\) where \((x, y)\) is its location in the image and \(\phi\) is its orientation. For example, the label of a positive image contains the poses of the head, torso and limbs of a human in the image (see Fig. 1).

Within this scenario, our goal is to obtain a set of parameters such that the model assigns a high energy to the object in the positive image and a low energy to all putative poses of the object in the negative images. In other words, the parameters distinguish between positive and negative examples.

#### 2.1. Parts-based Model

A parts-based model is a graph \(G = (V, E)\) whose vertices \(V\) correspond to parts and edges \(E\) describe pairwise linkages between parts. We restrict the edges of the graph to form a tree. This has two computational advantages: (i) the parts-based model can be efficiently matched to a test image using max-sum belief propagation (BP) [20]; and (ii) as will be seen shortly, it also allows us to learn the optimal parameters of the model efficiently.

We denote by \(Z\) a set of labels \(\{z_0, z_1, \cdots, z_{n-1}\}\) where each label represents a putative pose of a part, i.e. \(z_i = (x_i, y_i, \phi_i)\). Each part \(a \in V\) can take one label from the set \(Z\). A labelling of the model is defined by a function \(f(\cdot)\) such that \(f : V \rightarrow \{0, 1, \cdots, h - 1\}\), i.e. the part \(a\) takes a pose \(z_{f(a)}\). Note that there are \(h^n\) possible labellings, where \(n = |V|\) denotes the number of parts. In order to quantitatively distinguish between one labelling and the other, we assign the following energy to each labelling:

\[
Q(f) = \sum_{a \in V} \bar{g}_{a;f(a)} + \sum_{(a,b) \in E} \bar{g}_{ab;f(a)f(b)} + \kappa, \tag{1}
\]

with the best labelling taking the maximum energy. The terms \(\bar{g}_{a;f(a)}\) and \(\bar{g}_{ab;f(a)f(b)}\) are the unary and pairwise potentials respectively and \(\kappa\) is a constant. We assume the form of the potentials to be the following:

\[
\bar{g}_{a;f(a)} = w^T_a \theta_{a;f(a)}, \quad \bar{g}_{ab;f(a)f(b)} = w_{ab} \theta_{ab;f(a)f(b)}, \tag{2}
\]

where \(\theta_{a;f(a)}\) is the feature vector for part \(a\), computed using the image data defined by the pose \(z_{f(a)}\). For this work, we restrict the representation of each part to be a rectangle of fixed dimensions (see Fig. 1). The terms \(\theta_{ab;f(a)f(b)}\) are the pairwise features. Note that the feature vector \(\theta_{a;f(a)}\) represents the shape and appearance of each part, while the pairwise feature \(\theta_{ab;f(a)f(b)}\) represents the spatial configuration of a pair of neighbouring parts. The terms \(w_a, w_{ab}, \kappa\) denote the parameters of the parts-based model. Given a test image, the best match of the parts-based model (i.e. the pose of the object) is determined by running max-sum BP which finds the labelling \(f(\cdot)\) that maximizes the energy.

#### 2.2. Positive Training Data

We denote by \(D^p_+\) the \(k^{th}\) positive image, where \(k \in \{0, 1, \cdots, n_+ - 1\}\). Furthermore, let \(f^k(\cdot)\) define the poses
of the parts in the image. Each positive image provides one positive example which is represented as

$$\theta^k_{ab} = (\theta^{k}_{a,f_k(a)}, \forall a \in V; \theta^{k}_{ab,f_k(a)f_k(b)}, \forall (a, b) \in E),$$

(3)

where the operator (a) denotes vector concatenation. We use HOG [6] and skin colour [13] to define the feature vector $\theta^{k}_{a,f_k(a)}$. The pairwise features are represented using a Potts model which favours all valid pairwise configurations equally. A valid configuration is defined as one which lies close to a configuration observed in the positive examples. In other words, it lies within a small hypercube in the pose space whose center is specified by the relative pose $z_{f_k(a)} - z_{f_k(b)}$ for some $k \in \{0, 1, \ldots, n_+ - 1\}$. Let $\mathcal{L}_{ab}$ represent the set of valid configuration for $(a, b) \in E$. Formally, the pairwise feature is given by

$$\theta^{k}_{ab,f_k(a)f_k(b)} = \begin{cases} 1 & \text{if } (f_k(a), f_k(b)) \in \mathcal{L}_{ab}, \\ 0 & \text{otherwise}. \end{cases}$$

(4)

Note that, by our definition, $\theta^{k}_{a,f_k(a)} = \theta^{k}_{ab,f_k(a)f_k(b)} = 1$ for all $k \in \{0, 1, \ldots, n_+ - 1\}$. Similar to using tree-structured models, the choice of Potts model pairwise features has two computational advantages: (i) it allows us to efficiently match the model to the test image by using the distance transform techniques of [8]; and (ii) as will be seen, it allows us to further reduce the computational complexity of learning the parameters of the model. Note that similar pairwise features have been successfully applied for object detection [16].

2.3. Negative Training Data

We denote by $\mathcal{D}^l$ the $l$th negative image, where $l \in \{0, 1, \ldots, n_+ - 1\}$. Note that there are $h^n$ putative poses of the object in this image, each corresponding to a labelling $f(\cdot)$ (where $n$ is the number of parts and $h$ is the number of possible poses for each part). Each labelling $f(\cdot)$ defines a negative example given by

$$\theta^l_{-,f} = \{\theta^l_{a,f(a)}, \forall a \in V; \theta^l_{ab,f(a)f(b)}, \forall (a, b) \in E\},$$

(5)

where $\theta^l_{a,f(a)}$ is the feature vector for part $a$ at pose $f(a)$ in image $\mathcal{D}^l$ (computed using HOG and skin colour). The pairwise feature $\theta^l_{ab,f(a)f(b)}$ is given by the Potts model in equation (4), i.e. 1 if $(f(a), f(b)) \in \mathcal{L}_{ab}$ and 0 otherwise.

2.4. The Learning Problem

The problem of learning the parameters of the model can now be cast as obtaining a weight vector $w$ and scalar bias $\kappa$ which discriminate between the positive example $\theta^+_a$ and all the negative examples $\theta^l_{-,f}$. Formally, let

$$w = (w_a, \forall a \in V; w_{ab}, \forall (a, b) \in E),$$

(6)

where the concatenation is in the same order as in the positive and negative examples. The dimensionality of $w_a$ and $w_{ab}$ is equal to the dimensionality of $\theta^k_{a,f_k(a)}$ and $\theta^k_{ab,f_k(a)f_k(b)}$ respectively. It can be easily verified that within this setting the energy of a given example $\theta$ can be concisely written as $w^\top \theta + \kappa$. Ideally, the weight vector and bias should satisfy the following:

$$w^\top \theta^+_a + \kappa \geq 1, \forall k \in \{0, \ldots, n_+ - 1\},$$

$$w^\top \theta^l_{-,f} + \kappa \leq -1, \forall l \in \{0, \ldots, n_+ - 1\}, f(\cdot),$$

(7)

where each negative image $\mathcal{D}^l$ induces $h^n$ possible labellings $f(\cdot)$. In other words, the parameters should be able to completely discriminate between positive and negative examples in terms of their energy values. However, in practice it is not always possible to separate the data. Hence, the most discriminative weight vector and bias are learnt using the following optimization problem:

$$(w^*, \kappa^*) = \arg \min_{w, \kappa} \frac{1}{2} ||w||^2 + C(\sum_k \kappa^k + \sum_l \xi^l),$$

(8)

s.t. $w^\top \theta^+_a + \kappa \geq 1 - \xi^k, \forall k,$

$$w^\top \theta^l_{-,f} + \kappa \leq -1 + \xi^l, \forall l, f(\cdot),$$

$$\xi^k \geq 0, \forall k, \xi^l \geq 0, \forall l.$$

(9)

(10)

(11)

Here, $C \geq 0$ is a user-defined constant which specifies the tradeoff between the accuracy and regularization of the weight vector, the operator $|| \cdot ||$ represents the standard $\ell_2$-norm and variables $\xi^k$ and $\xi^l$ denote the hinge loss for positive and negative examples respectively.

As observed in [9], the above problem is convex. However, it cannot be solved efficiently since inequality (10) specifies $h^n$ constraints per negative image $\mathcal{D}^l$. In [9], the authors suggested the following iterative algorithm: (i) given the current estimate of $w$, find the labelling $f(\cdot)$ that maximizes $w^\top \theta^l_{-,f}$ for each negative image (using max-sum BP); and (ii) add the constraints associated with the labellings $f(\cdot)$ to the constraints of the previous iteration which resulted in a non-zero hinge loss $\xi^l$. Note that the bottleneck of the above strategy is the first step which involves running an inference algorithm for each image at each iteration. In this work, we get rid of this bottleneck and obtain the globally optimal solution to this problem by reducing it to an equivalent problem with polynomial number of constraints.

2.5. Efficient Reformulation

The main difficulty with solving the above problem is that inequality (10) specifies a large number of constraints (exponential in the number of parts, i.e. $h^n$ where $h$ is the number of parts and $h$ is the number of putative poses for each part). However, we will show how inequality (10) can be reduced to an equivalent set of $O(nh|\mathcal{L}_{ab}|)$ constraints. We begin by reformulating inequality (10) as

$$t^l + \kappa \leq -1 + \xi^l, \quad t^l \geq w^\top \theta^l_{-,f}, \forall f(\cdot),$$

(12)

i.e. $t^l$ is an upper bound on the set of values $w^\top \theta^l_{-,f}$. We now show that this upper bound can be specified by a polynomial number (specifically $O(nh^2)$) of constraints. For simplicity of explanation, let us assume that $\mathcal{E} = \{(a, b), \forall b \in V - \{a\}\}$, i.e. the model is star-shaped with part $a$ as the root. We note however that extending our arguments to a general tree-structured model is straightforward.

We define variables $M^l_{ba;i}$ using $h$ constraints such that

$$M^l_{ba;i} \geq w^\top_{b;i} \theta^l_{b;i} + w^\top_{ab} \theta^l_{ab;i}, \forall z_j \in Z,$$

(13)
where \( b \in \mathcal{V} - \{a\} \) and \( z_i \in \mathcal{Z} \). Since one \( M_{ba;i}^l \) is defined for each \((a, b) \in \mathcal{E} \) and \( z_i \in \mathcal{Z} \), the total number of constraints is \( O(nh^2) \). Note that the RHS of inequality (13) is the message that \( b \) passes to \( a \) when performing max-sum BP on the log-linear model defined by weights \( w \) and features \( \theta \). The upper bound \( t^l \) is specified by

\[
t^l \geq w^T_a \theta^i_{a;i} + \sum_{b \in \mathcal{V} - \{a\}} M^l_{ba;i}, \forall z_i \in \mathcal{Z}. \tag{14}
\]

Hence, inequality (10) can be replaced by inequalities (12), (13) and (14). Note that till now we have not used the fact that the pairwise features \( \theta^i_{ab;j} \) form a Potts model, i.e. the above reformulation is valid for any tree-structured model. However, it is well-known that the messages in max-sum BP can be computed more efficiently for a Potts model \([8]\). Equivalently, the slack variables \( M_{ba;i}^l \) can be specified using fewer constraints as follows:

\[
m^l_b \geq w^T_b \theta^i_{b;j} + w^l_{ab}, \forall z_j, t^l_{ab;i} \geq m^l_i, \tag{15}
\]

where \((i, j) \in \mathcal{L}_{ab}\) if and only if \( z_i \) and \( z_j \) specify valid configurations. Note that the number of constraints required to specify \( M_{ba;i}^l \) for all \( z_i \in \mathcal{Z} \) using inequality (13) is \( O(h^2) \). In contrast, for the Potts model, the same slack variables can be specified with \( O(h|\mathcal{L}_{ab}|) \) constraints using inequalities (15), where \(|\mathcal{L}_{ab}| \ll h \) (since typically only a small fraction of pairwise configurations are valid ones). Thus, we follow the following efficient convex problem (with \( O(h|\mathcal{L}_{ab}|) \) constraints per negative image) which provides the optimal weight vector and bias for the given training data:

\[
(w^*, \kappa^*) = \arg \min_{w, \kappa} \frac{1}{2} ||w||^2 + C \left( \sum_k \kappa^k + \sum_l \kappa^l \xi^l \right), \tag{18}
\]

s.t. \( w^T \theta^i_a + \kappa \geq 1 - \xi^k, \kappa \geq 0, \forall k, \)

\[
t^l + \kappa \leq -1 + \xi^l, \kappa^l \geq 0, \forall l, \tag{16}
\]

\[
m^l_b \geq w^T_b \theta^i_{b;j} + \sum_l M^l_{ba;i}, \forall l, i, \forall z_j, \tag{15}
\]

\[
m^l_b \geq w^T_b \theta^i_{b;j}, \forall l, i, \tag{16}
\]

\[
m^l_{ba;i} \geq w^l_{ab}, \forall l, i, \tag{15}
\]

The above reformulation is done in the primal domain itself. In contrast, the work of [24] performs all the changes in the dual and hence, does not allow us to take advantage of the Potts model pairwise features. It is worth noting that the above trick of reducing the number of constraints can also be applied to other commonly used pairwise features such as (truncated) linear, and (truncated) quadratic models (using the distance transform technique of \([8]\)).

### 2.6. The Dual Problem

As with SVMs, the above primal problem has an interesting dual which (i) can be solved efficiently; (ii) allows the use of the kernel trick; and (iii) directly provides the optimal weight vector. Recall that, given the optimization problem

\[
\min g_0(x), \; \text{s.t.} \; g_l(x) \geq 0, \; l = 1, \ldots, m, \tag{17}
\]

its dual is defined as

\[
\max_{\alpha \geq 0} L(x, \alpha), \; L(x, \alpha) = g_0(x) - \sum_i \alpha_i g_i(x), \tag{18}
\]

where the function \( L(\cdot) \) is called the Lagrangian and the variables \( \alpha_i \) are called the Lagrange multipliers. Using the Karush-Kuhn-Tucker (KKT) conditions, the dual of problem (16) can be simplified to

\[
\max \sum_k \alpha^k - \frac{1}{2} \sum_{(a, b) \in \mathcal{E}} \alpha^a_{ab} (V_{ab} V^T_{ab} \bullet y_{ab} y^T_{ab}) \alpha^a_{ab} + \sum_l \alpha^l - \frac{1}{2} \sum_{(a, b) \in \mathcal{E}} \alpha^l_{ba} (V_{ba} V^T_{ba} \bullet y_{ba} y^T_{ba}) \alpha^l_{ba}
\]

s.t. \( 0 \leq \alpha^k \leq C, 0 \leq \alpha^l \leq C, \)

\[
\sum_k \alpha^k_{ab} - \sum_l \alpha^l_{ba} = 0, \alpha^l_{ba} = \sum_i \alpha^i_{ab;i}, \alpha^l_{ab;i} = \sum_j \alpha^j_{ab;j}, \tag{19}
\]

where \((\bullet)\) represents the Frobenius inner product and \( \alpha_{pq} = (\alpha^k_{pq}, \forall k; \alpha^l_{pq;i}, \forall l, i; \alpha^l_{pq;j}, \forall l, j) \).

The matrix \( V_{pq} \) is given by

\[
\begin{pmatrix}
\left( \frac{1}{d_p} \partial_{pq} f^k(p) \cdot \frac{1}{d_{pq k}} f^k(q) \right)^T, \forall k
\
\left( \frac{1}{d_p} \partial_{pq i} \right)^T, \forall l, i
\
\left( \frac{1}{d_p} \partial_{pq j} \right)^T, \forall l, i, j
\end{pmatrix}, \tag{21}
\]

where \( d_p \) is the degree of part \( p \). Since problem (16) is a convex quadratic program, strong duality holds. In other words, its optimal value is the same as the optimal value of problem (19). The dual problem uses only the dot product of various features (i.e. \( V_{pq} V^T_{pq} \)). Hence, it allows us to use the kernel trick which projects the features to a higher dimensional space. However, in this work we will restrict ourselves to the linear kernel since our features, in particular HOG, have been shown to perform well using this setting \([6]\). Note that the KKT conditions provide the optimal weight vector in terms of the dual solution \([29]\). Hence, instead of working in the primal domain, we can obtain the parameters of the model by solving only the dual problem.

Note that problem (19) is similar in flavour to the well-known SVM learning problem \([29]\). However, standard SVM softwares such as SVM-Light \([14]\), which update only a small number of Lagrange multipliers at each iteration, cannot be directly used to obtain its optimal solution. This is due to the fact that, while the minimal optimization problem of an SVM consists of two Lagrange multipliers, problem (19) has a very large minimal optimization problem. Recall that the size of the minimal optimization problem is the minimum number of variables that are required to move to a different solution from the current one without violating any of the constraints. For example, consider two Lagrange multipliers \( \alpha^i_{ba;j} \) and \( \alpha^j_{ba;i} \), where \( l \neq l' \). These two variables cannot be changed without violating the constraints

\[
\sum_i \alpha^i_{ab;i} = \sum_j \alpha^j_{ba;j}, \forall l. \tag{22}
\]
In other words, the size of the minimal optimization problem of (19) is a lot bigger than 2 due to the presence of the above constraint. In order to overcome this issue, we design an algorithm specifically for solving problem (19) as described in the next section.

3. Learning the Optimal Model

We now describe how problem (19) can be solved efficiently using a method based on dual decomposition. We begin by outlining the general dual decomposition approach and later provide the details for our problem.

3.1. Dual Decomposition

Dual decomposition is a powerful technique which allows us to efficiently solve several optimization problems [2]. It has recently been successfully used for MAP estimation of random fields [15] and for obtaining feature correspondences [25]. To describe dual decomposition, we consider the following convex optimization problem:

$$
\min_x \sum_{i=1}^m g_i(x), \text{ s.t. } x \in C,
$$

(23)

where $C$ represents the convex feasible region of the problem. The above problem is equivalent to the following:

$$
\min_{x_i,x} \sum_i g_i(x_i), \text{ s.t. } x_i \in C, x_i = x,
$$

(24)

which allows us to obtain its Lagrangian dual as

$$
\max_{\lambda_i} \min_{x_i \in C} \sum_i g_i(x_i) + \sum_i \lambda_i(x_i - x).
$$

(25)

Differentiating the dual function with respect to $x$, we obtain the constraint that $\sum_i \lambda_i = 0$. Substituting this above dual problem, we can further simplify the dual as

$$
\max_{\sum_i, \lambda_i = 0} \min_{x_i \in C} \sum_{i=1}^m (g_i(x_i) + \lambda_i x_i).
$$

(26)

The above form of the dual suggests the following strategy for solving it. We start by initializing $\lambda_i$ such that $\sum_i \lambda_i = 0$. Keeping the values of $\lambda_i$ fixed, we solve the following slave problems (i.e. subproblems of the above dual):

$$
\min_{x_i \in C} (g_i(x_i) + \lambda_i x_i),
$$

(27)

where $g_i(\cdot)$ should be chosen such that it can be optimized exactly and efficiently. Upon obtaining the optimal solutions $x_i^*$ of the slave problems, we update the values of $\lambda_i$ by projected gradient descent where the gradient with respect to $\lambda_i$ can easily be verified to be $x_i^*$. In other words, we update $\lambda_i \leftarrow \lambda_i + \eta \nabla g_i(x^*)$ where $\eta$ is the learning rate at iteration $t$. In order to satisfy the constraint $\sum_i \lambda_i = 0$ we project the value of $\lambda_i$ to $\lambda_i \leftarrow \lambda_i - 1/m \sum_i \lambda_i$ where $m$ is the number of slave problems. Under fairly general conditions, this iterative strategy known as dual decomposition can be shown to converge to the globally optimal solution of the original problem (23). We refer the reader to [2] for details.

The critical component of any dual decomposition approach is the choice of the slave problems. Clearly, the choice should be such that problem (27) can be solved quickly. In what follows, we specify the appropriate slaves for problem (19) which can be optimized efficiently.

3.2. The Slave Problems

We consider slave problems of the following form:

$$
\max \sum_k \alpha_k^i + \sum_{t,i} (\alpha_{t,i}^l + \sum_j \alpha_{t,ij}^l) - \frac{1}{2} \alpha_q^T (V_{pq} V_{pq}^T \cdot y_{pq} y_{pq}^T) \alpha_{pq}
$$

s.t. $0 \leq \alpha_k^i \leq C, 0 \leq \sum_i (\alpha_{t,i}^l + \sum_j \alpha_{t,ij}^l) \leq C$,

$$
\sum_k \alpha_k^i - \sum_{t,i} (\alpha_{t,ij}^l + \sum_j \alpha_{t,ij}^l) = 0.
$$

(28)

Specifically, for each edge $(a, b) \in E$, we define two slave problems: (i) where $p = a, q = b$; and (ii) where $p = b, q = a$. The main difference between the slaves and problem (19) is the absence of the constraint (22). Recall that this constraint was the reason for a large minimal optimization problem for (19) (see last paragraph of § 2.6). By removing this constraint and enforcing it implicitly through variables $\lambda_i$, we are able to obtain slave problems which are easier to solve due to two reasons: (i) each slave problem is defined using $O(n_p + h_n) L_{ab}(n_m)$ variables instead of $O(n_p + h_n) \sum_{(a,b) \in E} |L_{ab}(n_m)|$ variables in problem (23); and (ii) similar to the SVM optimization problem, the size of the minimal optimization problems of the slaves is 2. In fact, the only difference between SVM and the above slave problems is that instead of each Lagrange multiplier being bounded by the parameter $C$, the sum of all the Lagrange multipliers corresponding to the same negative image is bounded by $C$, i.e. $0 \leq \sum_i (\alpha_{t,ij}^l + \sum_j \alpha_{t,ij}^l) \leq C$.

We observe that each slave corresponds to the problem of obtaining the weights for the feature vector of a part together with its pairwise feature with a neighbouring part. Using the fact that the weight for the pairwise feature is a scalar (which can either be positive or negative), we can greatly reduce the number of variables present in the slave. Specifically, for each negative image $D_{ij}$ and putative pose $p$ of part $q$, we only need to consider two poses of part $q$: one that defines the maximum pairwise feature with pose $z_{ij}$ of $p$ and one that defines the minimum pairwise feature (since $\alpha_{t,ij}^l = 0$ for all other poses $z_j$). In other words, the effective number of variables for each slave problem is $O(n_p + h_n)$ which is equal to the number of variables when each part is learnt individually. In our implementation, we modified the highly optimized SVM-Light software [14] to solve the slave problems. It can be verified that all the steps of SVM-Light can be implemented efficiently for our case. We omit the details due to lack of space.

4. Implementation Details

Features. For a particular pose of a part, we represent its shape using the variation of HOG described in [9] (using
the authors’ code) where histograms are computed on non-overlapping rectangular cells. The dimensionality of each histogram is fixed to 9. The histogram of a particular cell is normalized over the 4 possible $2 \times 2$ blocks of cells which contain that cell. In our experiments, we use cells of size $24 \times 24$ pixels for the largest part (i.e. the torso of a human) and $16 \times 16$ pixels for all other parts. In order to represent the appearance of a part, we train a linear SVM for skin colour detection using RGB values (normalized to sum to 1) of a pixel. Then, given a pose of a part, we want appearance to contain that cell.

Let $w$ denote this fraction. The appearance at the pose is modelled as the appearance has the advantage of favouring a certain fraction of pixels belonging to skin (since we learn a weighting $w_1x + w_2x^2$). In contrast, using just $x$ as the appearance feature would only allow us to either favour skin pixels or not (depending on whether $w_1$ is positive or negative).

Positive and negative data. As mentioned before, the user provides the poses of the parts within the positive images. We consider all other poses of the parts in the positive images to be negative examples (i.e., our positive and negative images are the same, only the poses of the parts are different). In order to obtain the putative poses of the parts, we train a linear SVM for each part of the object using the positive examples of the part together with randomly sampled negative examples (whose overlap with the positive examples is less than a certain threshold). The putative poses of the part in a given image are then defined as the poses that result in the $h$ top matches (i.e., with the highest score) using the linear SVM for that part.

Occlusion. In order to handle the occlusion of parts, we provide two options for each putative pose: (i) the part takes that pose and is visible; and (ii) the part takes that pose and is occluded. This effectively doubles the number of possible poses of each part. The unary potential of part $a$ being occluded is modelled using a constant $w^0_a$. For a pair of neighbouring parts $(a, b) \in E$, the pairwise potentials for only part $a$ being occluded, only part $b$ being occluded and both parts being occluded are modelled as constants $w^{ab}_a$, $w^{ab}_b$ and $w^{ab}_{ab}$ respectively. All the above constants are learnt within the framework described in this paper.

Memory requirements. The slave problems (28) are defined over points that have a great deal of redundancy, e.g., see equation (21) where several points have $\theta_{l_{p,i}}$ in common. Hence, we effectively have to store only the feature vectors of the parts in positive and negative examples together with the pairwise features for pairs of poses consider in problem (28). This requires only slightly more memory than (and can be initialized using) a linear SVM trained on the feature vectors of individual parts. In all of our experiments, the input to the slave problems fitted in the main memory. Hence, we were able to solve problems (28) efficiently using the modified SVM-Light algorithm.

5. Results

We now describe our experimental setup and the results obtained using two challenging datasets.

Sign language dataset. In the first experiment, we use 5 videos of sign language. For each video, 39 frames have been manually labelled with the poses of 8 parts which constitute the human upper body [4]

\[ \text{http://www.robots.ox.ac.uk/~vgg/data/sign_language} \]

Fig. 1 shows the tree-structured model and an example image for this dataset. We use 100 frames (20 from each of the 5 videos) as training data and the remaining 95 frames as test data. The training data is used to learn the parameters of the tree-structured model. For this dataset, the number of parameters to be learnt is $5777$ (4 for each of the 7 edges $E$, and the rest for the shape and appearance features of the parts). The learnt tree-structured model is then matched to each of the test images using the standard max-sum BP algorithm together with the distance transform techniques of [8]. This provides us with the most likely pose of the parts in each frame. In order to quantitatively evaluate the results, we compute the overlap score for each part in each test frame. The overlap score is computed as the ratio of the area of the intersection of the ground truth with the part found using the model to the area of their union. A part is said to be correctly localized if the overlap score is greater than 0.25. The accuracy of the learnt model is measured as the percentage of parts that are correctly localized in all the test images.

We compare our method with two iterated SVM (ISVM) strategies using different number of putative poses $h$. The first, which we call ISVM-1, is the coordinate-descent style approach adopted in [18] for solving optimization problems with a convex objective function and bilinear constraints. Specifically, given the current weight vector $w$, the best negative pose of the object is obtained in each image. A new $w$ is then computed using only these negative examples to-
gether with the positive examples. The second approach, ISVM-2, is the method used by [9] adapted to the supervised case. Specifically, the poses of the parts in the positive examples are fixed to their ground truth value thereby making the approach similar to the learning algorithm for structured output regression [27]. ISVM-2 differs from ISVM-1 in that at each iteration the negative examples computed in all the previous iterations which result in a non-zero hinge loss are also included within the optimization problem. Note that both the ISVM strategies attempt to solve the same optimization problem as our approach, i.e. equations (8)-(11). In other words, the experimental setting for all the methods is the same thus providing a fair comparison, both in terms of efficiency and accuracy, for our approach.

In all the experiments, our method converges quickly to the global optimal (between 0.5 to 2 days, depending on the value of $h$, on a 3GHz processor with 4 GB RAM). However, both ISVM-1 and ISVM-2 are computationally inefficient. Hence, we are unable to run them till convergence and instead, stop after twice the amount of time taken by our approach. The reason for the inefficiency of ISVM-1 and ISVM-2 is as follows. It takes between 3 and 15 seconds to perform inference on each training image. Although 15 seconds is fairly quick, it implies that each iteration of ISVM takes about half an hour. According to [27], the number of iterations required is $O(n_1)$ where $n_1$ is the number of training images. In other words, it takes hundreds of hours (i.e. several days) to reach convergence. In contrast, our dual decomposition strategy converges when all the slaves have the same optimal solution. Consider two slave problems which share some common variables. Such slaves are defined on similar points (specifically, the feature vectors of the part for positive and negative examples are the same in both the slave problems). Furthermore, they are initialized to the same good solution using a linear SVM trained only on the individual parts (see section 4). Hence, they converge to a similar optimal solution within 2 to 3 iterations of the dual decomposition. As a result, our method provides the globally optimal model more efficiently than ISVM-1 and ISVM-2. Put another way, the main reason for the inefficiency of our method is that we replace the inference problem in ISVM by that of choosing the right subproblem of the slave problems at each iteration. Similar to SVM-Light, this choice is made based on the first order approximation of the corresponding objective function which can be computed very quickly [14].

Table 1 shows the accuracy of the three approaches using different number of putative poses $h$ during training. Our method converges to the globally optimal solution efficiently, thereby providing the best pose estimation results. We also computed the accuracy of our approach on the test dataset employed in [4] using their evaluation measure. Note that [4] used a training data which consisted exclusively of the signer in the test data. In contrast, our training data consisted of 5 signers, none of whom were present in the test data. In other words, our experimental setting is much harder than [4]. However, our accuracy of 86.37% compares favourably to the 87.7% obtained by [4]. Fig. 2 shows some of the results with the poses of the parts overlaid on the test images. In each case, the model was matched to a frame in 2 – 3 minutes on average. During testing tens of thousands of putative poses per part are used, as compared to training where only a few hundred poses are employed. Hence, the cost of inference during testing is much higher. However, note that this cost is the same for all models, regardless of how they are learnt, since they all have to account for the articulation of the object (unlike [9] where rotation of parts is not considered).

**Buffy dataset.** In the second experiment, we use manually labelled frames from 4 episodes of the TV series ‘Buffy the Vampire Slayer’ [11]. The tree-structured model consists of six parts (since the hands are not labelled) which are connected together in a similar manner to the model shown in Fig. 1(a). The 196 frames from the first two episodes are used as training data for three approaches: our method, ISVM-1 and ISVM-2. The total number of parameters to be estimated is 4503 (4 parameters for each of the 5 edges $E$, and the rest for the shape and appearance features of the parts). Once again, our method converges faster than ISVM-1 and ISVM-2 (which are allowed to run for twice as long). The accuracy of the methods is evaluated using the same

<table>
<thead>
<tr>
<th></th>
<th>$h=250$</th>
<th>$h=500$</th>
<th>$h=1000$</th>
<th>$h=1500$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our</td>
<td>49.02</td>
<td>61.05</td>
<td>75.79</td>
<td>78.95</td>
</tr>
<tr>
<td>ISVM-1</td>
<td>3.16</td>
<td>7.54</td>
<td>6.49</td>
<td>7.01</td>
</tr>
<tr>
<td>ISVM-2</td>
<td>41.05</td>
<td>45.09</td>
<td>48.42</td>
<td>52.28</td>
</tr>
</tbody>
</table>

Table 1. The accuracy of three different approaches on the sign language dataset. Note that ISVM-1 is slow and susceptible to local minima and hence, provides inaccurate results. The ISVM-2 algorithm takes a long time to converge since it performs inference at each iteration. Despite running ISVM-1 and ISVM-2 for twice as long as our method, our model obtains the most accurate results.

---

3Available at http://www.robots.ox.ac.uk/~vgg/data/stickmen.
Our

ISVM-1

ISVM-2

Table 2. The accuracy of three different approaches on the Buffy dataset. Similar to the first experiment, our method provides the most accurate test results for all values of $h$ used during training. Scoring mechanism as the first experiment (i.e. we compute the overlap score for each part in each image, and then count the percentage of correctly localized parts) using the 204 frames of the last two episodes. Fig. 3 shows some example poses of the object in test frames. Table 2 lists the accuracy of the three approaches. Similar to the first experiment, our method learns the optimal model efficiently and provides the most accurate test results. Compared to the method of [11], which obtain accuracies of 56%, our model provides an accuracy of 39.2% using their evaluation criterion. However, unlike [11] our approach does not use the information provided by colour features and graph cuts based segmentation. Without these additional cues, [11] obtain an accuracy of 41% which is comparable to our approach.

<table>
<thead>
<tr>
<th></th>
<th>$h=250$</th>
<th>$h=500$</th>
<th>$h=1000$</th>
<th>$h=1500$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our</td>
<td>28.92</td>
<td>31.00</td>
<td>32.59</td>
<td>48.04</td>
</tr>
<tr>
<td>ISVM-1</td>
<td>2.45</td>
<td>2.94</td>
<td>8.33</td>
<td>9.80</td>
</tr>
<tr>
<td>ISVM-2</td>
<td>19.61</td>
<td>22.06</td>
<td>24.26</td>
<td>31.37</td>
</tr>
</tbody>
</table>

6. Discussion

We designed an approach for solving the important problem of learning discriminative parts-based models which can be employed in a variety of tasks such as recognition, detection and pose-estimation. We tested our approach on learning human upper body using very challenging datasets. However, the methodology is applicable for learning any tree-structured model, articulated or non-articulated.

Acknowledgements. This work was funded by the PASCAL Network of Excellence, ERC grant VisRec agreement no. 228180, ONR MURI N00014-07-1-0182, and EPSRC grant EP/C006631/1(P). Philip Torr is supported by a Royal Society Wolfson Research Merit Award.

References