Multiple Component Learning for Object Detection

Piotr Dollár[1,2] Boris Babenko[2] Serge Belongie[1,2] Pietro Perona[1] Zhuowen Tu[3]

[1] Electrical Engineering
California Inst. of Tech.
{pdollar,perona}@caltech.edu

[2] Comp. Science & Eng.
Univ. of CA, San Diego
{bbabenko,sjb}@cs.ucsd.edu

[3] Lab of Neuro Imaging
Univ. of CA, Los Angeles
zhuowen.tu@loni.ucla.edu

Abstract. Object detection is one of the key problems in computer vision. In the last decade, discriminative learning approaches have proven effective in detecting rigid objects, achieving very low false positives rates. The field has also seen a resurgence of part-based recognition methods, with impressive results on highly articulated, diverse object categories. In this paper we propose a discriminative learning approach for detection that is inspired by part-based recognition approaches. Our method, Multiple Component Learning (MCL), automatically learns individual component classifiers and combines these into an overall classifier. Unlike previous methods, which rely on either fairly restricted part models or labeled part data, MCL learns powerful component classifiers in a weakly supervised manner, where object labels are provided but part labels are not. The basis of MCL lies in learning a set classifier; we achieve this by combining boosting with weakly supervised learning, specifically the Multiple Instance Learning framework (MIL). MCL is general, and we demonstrate results on a range of data from computer audition and computer vision. In particular, MCL outperforms all existing methods on the challenging INRIA pedestrian detection dataset, and unlike methods that are not part-based, MCL is quite robust to occlusions.

1 Introduction

Computer vision has recently witnessed rapid development in the areas of category detection (detection and localization) and recognition (categorization). In detection, approaches that learn classifiers from simple low-level features and large amounts of data can achieve low false positive rates for quasi-rigid objects [1–3]. In recognition, part-based methods, particularly patches-as-parts approaches [4–6], have proven effective at differentiating highly articulated categories. In this paper we propose a detection algorithm that is inspired by both lines of work. Through this combination, our system learns a robust, component-based classifier that achieves low false positive rates on articulated objects.

Our approach is based on training a discriminative set classifier through a combination of boosting and weakly supervised learning, specifically Multiple Instance Learning (MIL). The resulting method, Multiple Component Learning (MCL), learns individual component classifiers and combines these into an overall

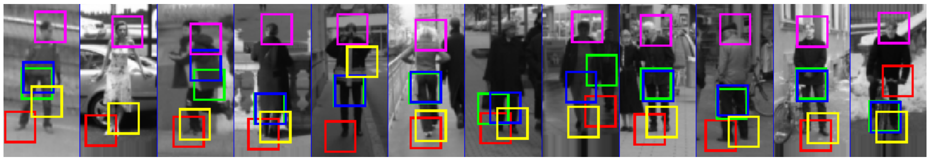

Fig. 1. Response of first 5 learned components classifiers on randomly selected INRIA pedestrian test images (best viewed in color). At most one box is displayed per component after non-maximal suppression and thresholding. Three components correspond to semantically meaningful parts (head-magenta, left foot-red, right foot-yellow); 2 correspond to the region between between the legs. The components were learned with *no component labels* provided during training. See Sec. 4.3 for details.

object classifier. Object labels in the form of bounding boxes are provided, but component labels are not. The approach is general, and we demonstrate results on a range of data from computer audition and computer vision, including state of the art results on the challenging INRIA pedestrian detection dataset.

We begin with a review of related work below, followed by a thorough description of our method in Sec. 2. Next we present a theoretical justification of our approach in Sec. 3, where we also review related work from the machine learning literature. Finally, we present experimental results in Sec. 4.

1.1 Related Work

A number of discriminative detection systems that learn from simple low-level features and large amounts of data have been proposed; typically their focus is either on the learning aspect [1, 3] or the design of appropriate features [2]. These methods require large amounts of labeled data to learn invariance to articulation, occlusion and intra-class variations. As mentioned, they have proven particularly successful for detecting rigid objects, achieving low false positive rates. MCL is similar to these methods in some regards, the key difference being that built into MCL is the domain knowledge that objects are composed of parts, that the mutual position of parts is somewhat variable and that parts may not all be visible. MCL is therefore much better suited for detecting articulated objects, *e.g.* pedestrians, and remains robust in the presence of occlusion.

Part-based approaches have a rich history; one of the earliest approaches dates back to 1973 [7]. A number of different ways of extracting parts from images have since been proposed. One approach involves designing part detectors by hand [8] or providing a system with labeled part examples [9, 10]. Other than the obvious disadvantage of being labor intensive, these methods are restricted to using a limited number of possibly sub-optimal parts. An alternative approach involves searching for repeatedly occurring elements using different criteria such as frequency of appearance in the training data [11], lowering an empirical risk function [12], or increasing mutual information [13]. Unlike in MCL, the recurring elements in these methods are fairly simple, including edge fragments [11], Gaussian models [12], or image fragments [13].

A simple but effective method of extracting parts is to crop small image patches, either using an interest point operator [4, 14] or by dense sampling [6]. The patches can be vector quantized to form 'codebooks' [4], and an image can then be represented using a 'bag of words' model [6]. Spatial information can be encoded using pairwise relationships [14] or with a spatial voting scheme [15]. Alternatively, a generative model can be used to model the object, *e.g.* the constellation model and its variants [16, 4, 17] have proven robust and capable of operating with little training data. Though effective in recognition tasks, these patch-based methods are limited by simple or fixed part appearance models, often relying on a patch distance measure in a predefined feature space, *e.g.* using normalized correlation or other patch descriptors. In MCL the component classifiers are learned from low level features, and although MCL requires more training data, the approach can lead to higher accuracy models.

Most closely related to our work is [18], which uses a formalism called latent SVMs to simultaneously learn part and object models. The resulting system is effective, though very different from our own. We emphasize that, as far as we are aware, aside from [18] MCL is the first part-based method that uses rich part appearance models without relying on part labels during training.

2 Multiple Component Learning

There are three primary challenges that we address in order to come up with our discriminative component-based object model. The first of these is how to learn a component classifier when only an object label is given. The second is how to learn *diverse* component classifiers given a method for learning a single component classifier. Finally, given multiple diverse component classifiers, we must combine these properly into an overall classifier for the object of interest. In this work we present a unified and effective solution to these challenges.

In order to learn component classifiers we turn to weakly supervised learning methods developed for object detection, where positive training images contain the object of interest, but, unlike the fully supervised case, the object location in each image is unknown [4, 17, 19]. Observe that learning a component classifier from images of objects where the components are in unspecified locations is an analogous problem. Thus, we can use weakly supervised learning to learn a single component classifier; specifically we use multiple instance learning (MIL) [20].

To learn a diverse collection of component classifiers we turn to boosting [21]. In boosting, multiple weak learners, each of which may have fairly high error, are combined into a single strong classifier with a low overall error. Weak classifiers are trained sequentially with the weights of the training samples adjusted so that incorrectly classified examples receive more weight. Boosting is ideally suited both for learning diverse classifiers and combining them into an overall classifier. The key to our approach is that we use component classifiers, trained using weakly supervised learning, as the weak classifiers in a boosting framework.

We refer to the individual constituent classifiers learned with weakly supervised learning as components rather than parts, as they need not always

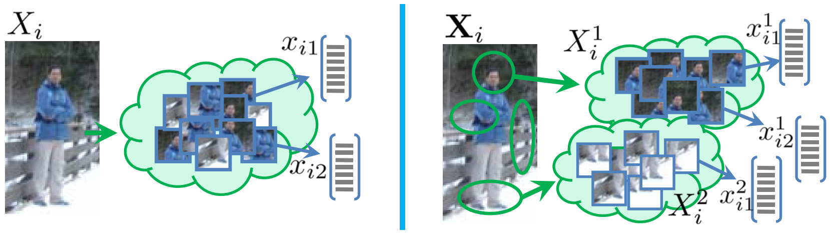

Fig. 2. We model an image as a *set* or *sequence of sets* of patches. **Left:** A pedestrian represented as a *set* of m densely sampled patches. A feature vector x_{ij} is computed for each patch, forming a set $X_i = \{x_{i1}, \ldots, x_{im}\}$. **Right:** Pedestrians have limited articulation, so each component appears in a limited region of the image. Prior to training, p overlapping regions are randomly generated (details in Sec. 4). For each region we extract patches, generating an ordered *sequence of sets* $\mathbf{X}_i = [X_i^1, \ldots, X_i^p]$.

correspond to semantically meaningful object parts (see Fig. 1). We name our approach *Multiple Component Learning* (MCL) to emphasize that we learn a classifier composed of a number of constituent classifiers.

We begin by laying out the problem formulation and introducing some notation, followed by an overview of MIL in Sec. 2.2. The foundation of our work is in the development of a practical set classification algorithm in Sec. 2.3.

2.1 Problem Formulation & Notation

We assume that the image of an object is composed of multiple components, each of limited spatial extent, arranged in a particular pattern. For a given object, only some of its components will be visible, and their position, scale, rotation, *etc.* may vary. If a component is present and visible, there will be an image patch that contains it; the challenge is that, a priori, it isn't known which patches contain relevant components.

We model an object as a collection of patches, or more formally as a *set* of patches. To model spatial relationships, alongside the appearance of each patch we keep track of its location (details discussed in Sec. 4). For objects with limited articulation, each component can appear in a limited region of the image, and it becomes useful to model the image as an ordered *sequence of sets*, one set for each fixed image region. See Fig. 2 for an illustration.

We now review the framework for standard discriminative learning, then move on to discriminative learning for sets and sequences of sets. Formally, in *supervised learning* data samples are represented by feature vectors from some data space $\mathcal{X}$, *e.g.* $\mathcal{X} = \mathbb{R}^d$. The goal is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{Y} = \{0, 1\}$ in binary classification. The learner is given a training data set $\{(x_1, y_1) \ldots, (x_n, y_n)\}$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, and outputs a function f that predicts the labels of novel data points.

Supervised learning for *sets* can be defined in an analogous way. The training data has the form $\{(X_1, y_1), \ldots, (X_n, y_n)\}$, where each $X_i = \{x_{i1}, \ldots, x_{im}\}$,

$x_{ij} \in \mathcal{X}$ and $y_i \in \{0,1\}$. We emphasize that the X_i are sets, *i.e.* the elements of X_i are unordered, and we denote this by $X_i \in \mathcal{X}^m$. Although MCL allows for sets of varying cardinality, for notational simplicity we assume a fixed size m. The goal is to learn a function $\mathcal{F} : \mathcal{X}^m \rightarrow \{0,1\}$ that generalizes well to unseen data. Technically speaking, set learning is a special case of supervised learning with the data space being $\mathcal{X}^m$. For the case of *sequences of sets*, each training instance $\mathbf{X}_i$ represents an ordered sequence of p (possibly overlapping) sets $\mathbf{X}_i = [X_i^1, \ldots, X_i^p]$, where $X_i^k \in \mathcal{X}^{m_k}$ represents a set like before. The goal is to learn a function $\mathcal{F} : [\mathcal{X}^{m_1}, \ldots, \mathcal{X}^{m_p}] \rightarrow \{0,1\}$ that generalizes well.

2.2 Multiple Instance Learning

Here we give an overview of multiple instance learning (MIL), which will serve as the basis for learning a single component classifier. In MIL, training data is given in sets, just as in MCL (in MIL terminology 'bag/instance' are used in place of 'set/element'). Labels are defined for instances, and a set is labeled positive if any instance in the set is positive and negative otherwise. Using the notation from above, a MIL classifier has the following form:

$$F(X_i) = \begin{cases} 1 & \text{if } \exists j \text{ s.t. } f(x_{ij}) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The goal is to minimize the error of F on the training data. Note that F depends entirely on the *instance classifier* $f : \mathcal{X} \rightarrow \{0,1\}$. Thus, although the training data is given in sets X, the goal of MIL is to learn a classifier f that operates on instances. $F(X_i)$ is essentially the maximum of $f(x_{ij})$ over j. If f outputs a probability or confidence in $[0,1]$, then we can define a 'soft' version of F:

$$\hat{F}(X_i) = \mathit{softmax}_j(f(x_{ij})), \tag{2}$$

where 'softmax' is some differentiable real-valued approximation of the max over $[0,1]$. We use the model of MIL defined in Equation (2) throughout this paper.

Note that MIL can be considered a special case of set learning where the form of $\mathcal{F}$ is restricted to the form given in Equation (1). We summarize the difference between standard classification, MIL and MCL in Fig. 3.

The term MIL was originally coined by Dietterich *et al.* [20] in their study of drug activity prediction, and in earlier work, Keeler *et al.* [22] trained a digit recognition system using unaligned image data. More recently, Viola *et al.* [19] introduced a boosting variant of MIL called MIL-BOOST which is effective and robust. Throughout this work we use an extended version of MIL-BOOST described in [23]. Some of these extensions are necessary for MCL, *e.g.* weighted sets, others improve performance, *e.g.* the generalized mean softmax model.

2.3 MCL Derivation

We are now ready to present the details of MCL. Again, the idea is to use component classifiers, trained using MIL, as the weak learners in a boosting framework.

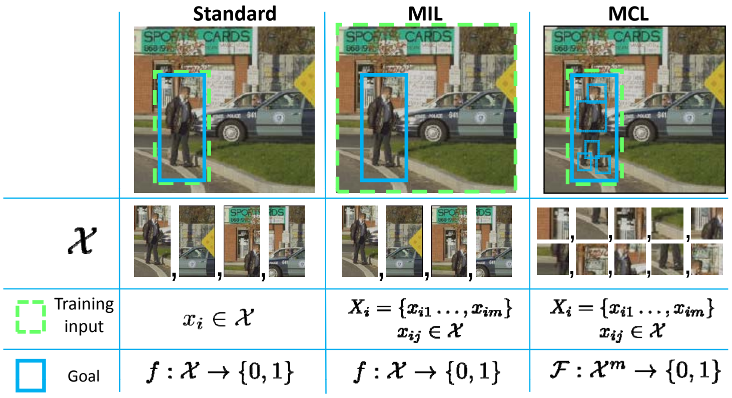
Standard
MIL
MCL
$\mathcal{X}$
Training input
Goal
$x_i \in \mathcal{X}$
$X_i = \{x_{i1} \ldots, x_{im}\}$ $x_{ij} \in \mathcal{X}$
$X_i = \{x_{i1} \ldots, x_{im}\}$ $x_{ij} \in \mathcal{X}$
$f : \mathcal{X} \rightarrow \{0,1\}$
$f : \mathcal{X} \rightarrow \{0,1\}$
$\mathcal{F} : \mathcal{X}^m \rightarrow \{0,1\}$

Fig. 3. Three learning paradigms, and how each applies to object detection. **Left:** Ground truth in supervised learning includes a label for each training instance. **Middle:** For MIL, labels are given only for sets, where a set is positive if it contains at least one positive instance, *e.g.* an image is labeled positive if the target object is present. Like in the standard case, the goal is to learn a function f that classifies instances. **Right:** In MCL data is likewise given in sets, but the goal is to learn a *set* classifier $\mathcal{F}$. Note that in this case $\mathcal{X}^m$ represents a set of candidate component patches.

We begin with a brief review of AdaBoost [21]. Given N labeled training examples (x_i, y_i) with $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, and an initial distribution $D_1(i)$, AdaBoost combines a number of weak classifiers h_t to learn a strong classifier $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$. Here $\mathcal{Y} = \{-1, 1\}$. Boosting has excellent generalization abilities and a strong theoretical foundation.

A close inspection of AdaBoost reveals that it makes no assumption about the form of the input space $\mathcal{X}$. The only requirement is that there exists a routine for training a weak classifier $h_t : \mathcal{X} \rightarrow \{-1, 1\}$ that has error less than $\frac{1}{2}$ for any arbitrary weighing D_t of the training data. Therefore AdaBoost can be trained on sets as long as the weak classifiers are suited for set classification.

We use MIL to train each weak classifier, or component, inside the boosting framework. Recall that MIL, given input data organized in sets $X_i \in \mathcal{X}^m$, learns a function $\hat{F} : \mathcal{X}^m \rightarrow [0, 1]$ of the form $\hat{F}(X_i) = \text{softmax}_j(f(x_{ij}))$. It is possible, using the extended version of MIL-BOOST, to train $\hat{F}$ using an arbitrary distribution over the training data [23]. Inserting $\hat{F}$ for h in AdaBoost, and adjusting the form of the input data to be in sets, along with some other details, gives rise to MCL-AdaBoost; see Fig. 4 for details.

Note that MIL is asymmetric and only learns components from the positive class; to also learn components from the negative class we can train MIL with the positive/negative labels swapped, and keep the better of the two components. Derivation of MCL with other boosting algorithms, such as RealBoost or GentleBoost [21], are similar to the derivation for MCL-AdaBoost. These require MIL to return real valued outputs, which is the case for MIL-BOOST. In our experiments

```
Given: $N$ labeled training examples $(X_i, y_i)$ with $y_i \in \{-1, 1\}$ and $X_i = \{x_{i1}, \ldots, x_{im}\}$, and initial distr. of weights $D_1(i) = \frac{1}{N}$ over the examples.

For $t = 1, \ldots, T$:
  • Train a MIL classifier $\hat{F}_t : \mathcal{X}^m \rightarrow [0, 1]$ using distribution $D_t$. Let $\hat{F}'_t(X_i) = (2 \times \mathbf{1}[\hat{F}_t(X_i) > th] - 1)$, where $th = .5$ or is chosen to minimize $\epsilon_t$.
  • Calculate error of $\hat{F}'_t : \epsilon_t = \sum_{i=1}^{N} D_t(i)\mathbf{1}(y_i \neq \hat{F}'_t(X_i))$.
  • Set $\alpha_t = -\frac{1}{2} \log(\epsilon_t / (1 - \epsilon_t))$.
  • Set $D_{t+1}(i) = D_t(i) \exp\left(-\alpha_t y_i \hat{F}'_t(X_i)\right) / Z_t$,
    where $Z_t = 2\sqrt{\epsilon_t(1 - \epsilon_t)}$ is a normalization factor.

Output the MCL classifier:    $\mathcal{F}(X) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t \hat{F}'_t(X)\right)$
```

Fig. 4. MCL-AdaBoost

MCL-RealBoost slightly outperforms MCL-AdaBoost; however, for simplicity we chose to use MCL-AdaBoost (which we abbreviate to MCL).

If each input is a *sequence of sets*, a simple variation of MCL proves effective. The training examples have the form $\mathbf{X}_i = [X_i^1, \ldots, X_i^p]$, where each X_i^k is a set. During each phase of boosting, for each value of $k < p$, we train MIL using X_i^k. Of the p candidates trained this way, we keep the one with lowest error and discard the rest. To make training more efficient, $T_1 < p$ classifiers can be selected instead. If $T_1 = T$, the diversity of the components comes from training each on different data as opposed to the reweighing step of boosting; however, boosting is still used to combine them. Details are omitted for space.

In the description of MCL given here, we assume that the input data has been organized into sets or sequences of sets and that MIL-BOOST is essentially a black box. Given this, MCL, like AdaBoost, only requires one parameter, the number of components T. In practice, we must also specify how to convert input data into sets, and the parameters to use when training MIL-BOOST, the most important being the feature space to use and the number of weak classifiers for MIL-BOOST. Moreover, the features used to capture spatial relations must be specified. These considerations are domain dependent, and we shall discuss them in more detail in the experiments sections.

3 Theoretical Foundations of MCL

Set learning can be approached as classifying distributions, where the elements of the set are observations sampled independently from an unknown density function. Examples include using statistical tests on histograms of quantized feature vectors to determine if two sets belong to the same distribution [6] and methods in computer audition that fit Gaussian Mixture Models (GMM) to the set distributions [24]. Alternatively, sets can be viewed as collections of elements where the order of the elements is unknown; however, an ordering is assumed to exist and part of the challenge is to recover it. A good example of this style of approach is kernel methods that implicitly or explicitly try to find correspondences between elements [25, 26].

Like many existing set learning algorithms, MCL assumes that an ordering of the elements in a set exists but is not known. If the ordering was given during training of MCL, the components could be learned using supervised learning as opposed to weakly supervised learning (see Fig. 4). However, unlike existing set learning algorithms, MCL does not require a distance measure between elements, nor does it assume the elements are generated from a simple model. Instead, learning occurs *both* at the instance level and at the set level. In this sense MCL is fundamentally different from existing work.

Here we give a more formal derivation of MCL. We begin by defining the problem of discriminative set classification in a way that does not rely on any assumptions at the instance level. The resulting formulation is intractable; however, imposing additional constraints leads to a more tractable approximation, resulting in MCL. This derivation will make explicit the assumptions made by MCL. It will help put MCL into context, and finally, it will allow us to explore how at least some of these assumptions could be lifted in the future.

A note on notation: the uppercase script letters $\mathcal{F}, \mathcal{G}$ refer to classifiers that operate on sets $X \in \mathcal{X}^m$ and lowercase letters f, g refer to classifiers over instances $x \in \mathcal{X}$, where typically $\mathcal{X} = \mathbb{R}^d$. There exist multiple approaches to learning *instance classifiers*, *e.g.* boosting, support vector machines, neural networks, *etc.* Here we describe set classifiers in terms of instance classifiers.

A set $X_i = \{x_{i1}, \ldots, x_{im}\}$ is unordered, so a set classifier $\mathcal{F}(X_i)$ should not depend on the indexing of the elements x_{ij}. Let $[x_{ij_1}, \ldots, x_{ij_k}]$ denote the object obtained by stacking k elements in the set X_i in fixed order $j_1, \ldots, j_k$ (with possible repetitions and omissions). A general form for a set classifier is:

$$\mathcal{F}^k(X_i) = \begin{cases} 1 & \text{if } \exists j_1, \ldots, j_k \text{ s.t. } f([x_{ij_1}, \ldots, x_{ij_k}]) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

In other words $\mathcal{F}^k$ is defined in terms of some instance classifier f, such that $\mathcal{F}^k(X_i) = 1$ iff there exists some ordering of the elements in X_i that satisfies $f([x_{ij_1}, \ldots, x_{ij_k}]) = 1$. Although fairly general, this form of $\mathcal{F}^k$ is not tractable for large k as computing $\mathcal{F}^k(X_i)$ involves a search. In the worst case, evaluating $\mathcal{F}^k(X_i)$ requires computing f over all m^k orderings of X_i of length k.

A crucial observation is that training $\mathcal{F}^k$ reduces to training a MIL classifier. For the case of $k = 1$ this reduction is direct since $\mathcal{F}^1(X) \equiv F(X)$, where $F(X)$ is the standard MIL formulation given in Equation (1). Thus we can use existing MIL approaches to learn the instance classifier f. The reduction to MIL for $k > 1$ is slightly more complex. Given a set X, we can exhaustively generate a set X^*, where each element in X^* is created by concatenating k elements $x_{j_1}, \ldots, x_{j_k} \in X$ into the vector $[x_{j_1}, \ldots, x_{j_k}]$. Close inspection of Equations (1) and (3) shows that $\mathcal{F}^k(X) \equiv F(X^*)$. Unfortunately this reduction is exponential in k, since by construction X^* contains m^k elements.

The smaller the value of k the more tractable learning and computing $\mathcal{F}^k$ becomes, but also the less information $\mathcal{F}^k(X_i)$ can use about X_i. To compensate for the loss of information from using small k, T different classifiers $\mathcal{F}_i^k$ can be combined using an instance classifier g. Define $\mathcal{G}^k$ as follows:

$$\mathcal{G}^k(X) = g\left(\mathcal{F}_1^k(X), \ldots, \mathcal{F}_T^k(X)\right) \tag{4}$$

The formulation of set learning given in Equation (3) is strictly more general than the form given in Equation (4); however, a potentially intractable search has been replaced with T smaller ones. Computing $\mathcal{G}^k(X)$ is $O(Tm^k)$, which is tractable for small k, and $\mathcal{G}^k$ can still depend on all elements in X regardless of k. Equation (4) thus represents a reasonable model for a set classifier.

MCL learns $\mathcal{G}^1$, with g being an additive model. Specifically, MIL and AdaBoost are used to sequentially learn and combine $\mathcal{F}_1^1, \ldots, \mathcal{F}_T^1$ into an overall set classifier. We now consider the approximations used to operationalize $\mathcal{F}^k$ into MCL. First, $\mathcal{G}$ is used as a tractable alternative to $\mathcal{F}$. Additive models have been shown to be robust and general [21], so their use for g is reasonable. With appropriate choice of a MIL training procedure, the components $\mathcal{F}_i^1$ are chosen from a rich hypothesis class and can thus be quite general. Perhaps the most severe approximation is use of $k = 1$ since each component $\mathcal{F}_i^1$ is limited to depend on a single instance. Further experiments could reveal if this is limiting in practice, *e.g.* for object detection. Assuming that m is not too large, deriving MCL using $k = 2$ or 3 would be straightforward given the reduction of $\mathcal{F}^k$ to MIL described above; however, it is outside the scope of this work.

4 Experiments

We experiment with data coming from three domains: (1) handwriting identification, (2) speaker identification and (3) pedestrian detection. We compile datasets for the first two domains; for the third we use the INRIA pedestrian dataset [2]. Finally, we discuss the role of data alignment in Sec. 4.4.

The first two experiments are meant to show the generality and power of MCL. We compare to two representative approaches. The first is AdaBoost [21] applied to a standard, not set-based, representation of the data. AdaBoost has access to identical features as MCL. We also compare to a simple bag of features (BOF) method: first we use k-means to quantize the feature space (k chosen via cross-validation), next we compute a k-bin histogram for each set, finally these histograms are input into AdaBoost. Learning occurs at the set level but not at the instance level (since the distances used for clustering are fixed).

4.1 Handwriting Identification

In the first experiment we apply MCL to the offline, text-independent handwriting identification problem (recognizing the author by his/her handwriting) [27]. Intuitively, a person's handwriting has a number of distinct features which are independent of the actual text, hence a set representation is appropriate.

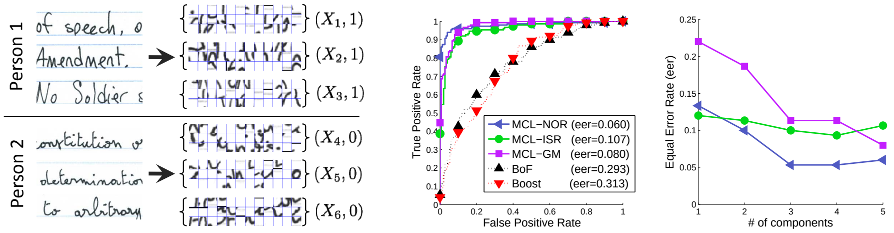

Fig. 5. Results for *writer identification*, see text for details.

Our data consists of images of 4 pages of text handwritten by 2 people. Each training example is a randomly sampled 70x50 window of text. For AdaBoost and MCL we used Haar features [1]; for BOF we use Euclidean distance on pixels to perform clustering. For each training example, 25x25 patches were randomly sampled to generate the sets. We trained MCL with three versions of MIL-BOOST, altering the softmax model (see Equation 2 and [23] for details). MCL significantly outperforms both AdaBoost and BOF. Results are shown in Fig. 5; the last plot shows performance as a function of number of components T.

4.2 Speaker Identification

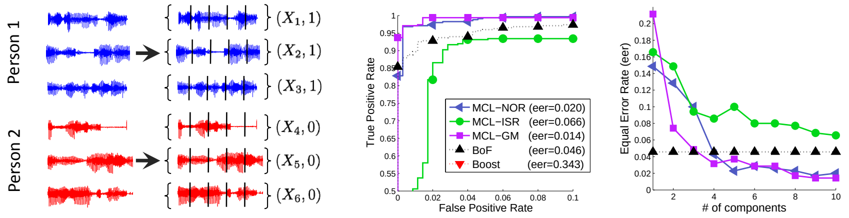

Fig. 6. Results for *speaker identification*, see text for details.

In this experiment our goal was to discriminate between the voices of John Kerry and George W. Bush (speaker identification is analogous to writer identification). We retrieved audio from a 2004 presidential debate[1] and cropped out roughly 6 minutes of each person speaking. The audio for each candidate was parsed into 360 second length clips, a quarter of which were used for training.

We used MFCC features which are commonly used in speaker recognition [24]. For AdaBoost one feature vector was generated per clip. For MCL and BOF, we generated a set representation of each clip by randomly sampling 78

[1] Available publicly at `http://www.archive.org/details/US_Pres_Debate_Sep.30.2004`

short windows, and computed 30-dimensional MFCC features for each window. AdaBoost performed very poorly (ROC curve is outside range of plot), while MCL with two variations of MIL-BOOST significantly outperformed BOF when trained with enough components T. Results are shown in Fig. 6.

4.3 Pedestrian Detection

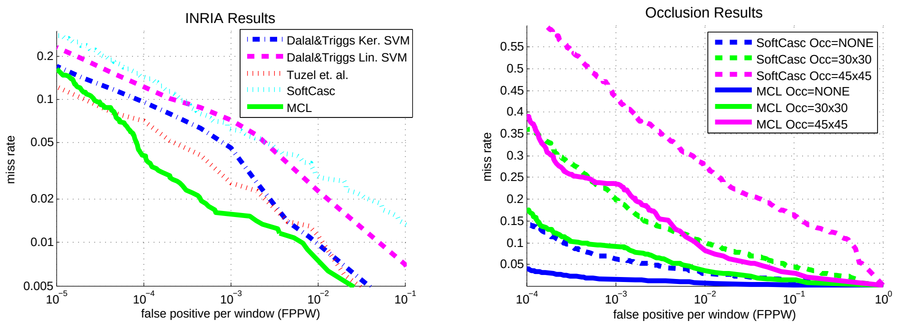

Fig. 7. Results on *pedestrian detection.* **Left**: MCL outperforms all reported results. At a false positive per window rate (FPPW) of 10^{-4}, a commonly used reference point, MCL has a miss rate of ∼4%, compared to ∼7% for [3] and ∼10% for [2]. For comparison we also implemented the *SoftCascade* approach described in [28], using the same candidate Haar features we use in MCL. Consistent with previously reported results, a cascade of Haars performs poorly. **Right**: Results on artificially generated occlusion. We overlaid random 30x30 or 45x45 patches into random locations in the pedestrian test images. We then regenerated the ROC curves for MCL as well as SoftCascade (which, like [2, 3], is not part-based). MCL on data with 30x30 performs similarly to SoftCascade on unoccluded data, and as the amount of occlusion increases, the gap between MCL and SoftCascade further increases.

We now present results on the INRIA pedestrian dataset [2]. A number of recent methods have targeted this data [2, 3, 29], We compare our results to each, using the training and evaluation methodology presented in [2], except to [29] as it appears the results reported in that work are inaccurate[2].

Here we provide details for training MCL on this dataset. Training windows are 128x64, and for each of these we extract ∼4K overlapping patches. We compute ∼10K random Haar features [1] per patch, using the original grayscale image, as well as gradient magnitude and 6 channels of gradient quantized by orientation. We use MIL-BOOST with the generalized mean softmax model to select 256 stump classifiers. All 2416 positive sets and ∼10K negative sets are used for training each MIL. We use the sequence of sets version of MCL (see Fig. 2).

[2] See `http://www.cs.sfu.ca/~mori/research/papers/sabzmeydani_shapelet_cvpr07.html`

Initially 50 binary masks in the shape of ellipses are randomly generated, and one MIL classifier is trained per mask, using sets generated from patches only in the masked region. The original binary masks are random and likely suboptimal; after training each MIL, we compute a new mask based on the MIL probability response images on the training positives and then retrain. This mask refinement step improves results. From among the 50 MIL candidate classifiers, AdaBoost is used to select the 20 best components. Next, we bootstrap ∼10K new negative windows from the training images from the false positives reported by MCL. The entire process is repeated 4 times to form a cascade. Our final MCL classifier is composed of 80 components (selected from 200 candidates).

We present a simple yet powerful method of incorporating spatial information into our MCL classifier. First, we compute the responses of the component classifier on the training images, giving one 128x64 probability map per training image for each component. Then, instead of using F (the max response over the set) as the weak classifier to AdaBoost, we use *Haar features computed over these probability maps.* Intuitively, single rectangle Haars capture absolute component location, multiple rectangle Haars capture relative component locations. Using these features, AdaBoost automatically learns the spatial relationships governing the components; no parameters need to be set in an ad-hoc manner. Using spatial information instead of the max was important for obtaining good results.

Training takes 2 weeks on a modern PC (majority of time is spent in MILBOOST training). Our classifier is composed of ∼20K Haar features (256 per MIL), compared to ∼6K in the Viola & Jones real-time face detector [1]; however, they are organized into multiple shallow cascades as opposed to a single deep cascade, so evaluation is slower. Simple feature sharing strategies should increase speed by an order of magnitude, reducing evaluation time to a few seconds per image.

The first five learned components are shown in Fig. 1. ROC curves comparing our method with Dalal and Triggs [2] and Tuzel *et al*. [3] are shown in Fig. 7, left. Part-based approaches have a natural advantage when occlusions are present. We show the robustness of MCL to occlusions in Fig. 7, right.

4.4 Role of Data Alignment

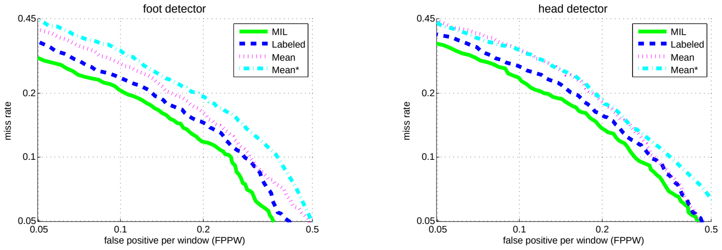

Fig. 8. Effects of alignment on training part classifiers, see text for details.

We perform a final experiment to show that using *aligned* data results in higher accuracy (part) classifiers. Note that simultaneous alignment of articulated objects is often impossible without relying on a part-based model.

We labeled the head and feet in the 2416 INRIA training pedestrians. For each part, we trained a classifier using patches sampled from each pedestrian in 1 of 3 ways: at the *labeled* part location, at the *mean* part location, and in a region around the mean part location (*mil*). We used AdaBoost for the first two cases and the extended version of MIL-BOOST for the third, using 256 Haars selected from an identical pool in each case. During testing, we applied each classifier to a region around the mean part location and recorded the maximum probability.

Results against bootstrapped negatives are shown in Fig. 8. Not surprisingly, *labeled* outperformed *mean*, showing alignment is beneficial. Additionally, *mean* performed better when we took the max probability over the region during testing rather than rely on the probability at the mean part location (*mean**). Also, consistent with the findings of [19], *mil* outperformed *labeled* even though it was trained in a weakly supervised manner (presumably our labeling is imperfect).

Together, these results make a strong case that data alignment is highly beneficial. As articulated objects typically cannot be fully aligned everywhere, MCL has an inherent advantage over methods that are not part-based.

5 Conclusion & Future Work

In this paper we presented a part-based object detection system called Multiple Component Learning (MCL). MCL combines elements of patch-based recognition and discriminative detection techniques to produce a high accuracy object detector that works well for articulated objects and is robust to occlusion. Unlike previous approaches, MCL learns powerful component classifiers without labeled part data. We showed results on data from computer vision and computer audition, including state of the art results in pedestrian detection.

In future work we plan on speeding up MCL, and seeing if tradeoffs in accuracy are necessary to achieve near real-time speeds. Additionally, results could be further improved by adapting MIL-BOOST to better suit the needs of MCL.

Acknowledgments

PD and BB were funded by NSF IGERT Grant DGE-0333451. SB was funded by NSF Career Grant #0448615 and the Alfred P. Sloan Research Fellowship. ZT was funded by NIH Grant U54RR021813 entitled Center for Computational Biology.

References

1. Viola, P., Jones, M.: Fast multi-view face detection. In: CVPR. (2001)
2. Dalal, N., Triggs, B.: Hist. of oriented gradient for human det. In: CVPR. (2005)
3. Tuzel, O., Porikli, F., Meer, P.: Human Detection via Classification on Riemannian Manifolds. In: CVPR. (2007)

4. Weber, M., Welling, M., Perona, P.: Unsupervised learning of models for recognition. In: ECCV. (2000)
5. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: CVPR. (2003)
6. Jurie, F., Triggs, B.: Creating efficient codebooks for vis. recog. In: ICCV. (2005)
7. Fischler, M., Elschlager, R.: The Representation and Matching of Pictorial Structures. IEEE Transactions on Computers **100**(22) (1973) 67–92
8. Brunelli, R., Poggio, T.: Face recog.: features vs. templates. PAMI **15**(10) (1993)
9. Mohan, A., Papageorgiou, C., Poggio, T.: Example-based object detection in images by components. PAMI **23**(4) (2001) 349–361
10. Mikolajczyk, K., Schmid, C., Zisserman, A.: Human Detection Based on a Probabilistic Assembly of Robust Part Detectors. In: ECCV. (2004)
11. Amit, Y., Geman, D.: A computational model for visual selection. Neural Computation **11** (1999) 1691–1715
12. Bar-Hillel, A., Hertz, T., Weinshall, D.: Object class recognition by boosting a part-based model. In: CVPR. (2005)
13. Vidal-Naquet, M., Ullman, S.: Object recognition with informative features and linear classification. In: ICCV. (2003)
14. Agarwal, S., Roth, D.: Learning a sparse repr. for object det. In: ECCV. (2002)
15. Leibe, B., Leonardis, A., Schiele, B.: Robust Object Detection with Interleaved Categorization and Segmentation. IJCV (2005) 1–31
16. Leung, T., Burl, M., Perona, P.: Finding faces in cluttered scenes using random labeled graphmatching. In: ICCV. (1995) 637–644
17. Crandall, D.J., Huttenlocher, D.P.: Weakly supervised learning of part-based spatial models for visual object recognition. In: ECCV. (2006)
18. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: CVPR. (2008)
19. Viola, P., Platt, J.C., Zhang, C.: Multiple instance boosting for object detection. In: NIPS. (2005)
20. Dietterich, T.G., Lathrop, R.H., Lozano-Perez, T.: Solving the multiple-instance problem with axis parallel rectangles. Artificial Intelligence (1997)
21. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Technical report, Stanford University (1998)
22. Keeler, J.D., Rumelhart, D.E., Leow, W.K.: Integrated segmentation and recognition of hand-printed numerals. In: NIPS. (1990)
23. Babenko, B., Dollár, P., Tu, Z., Belongie, S.: Simultaneous learning and alignment: Multi-instance and multi-pose boosting. Technical Report CS2008, UCSD (2008)
24. Reynolds, D., Rose, R.: Robust text-indep. speaker ident. using gaussian mixture speaker models. In: IEEE Trans. on Speech and Audio Processing. (1995) 72–83
25. Grauman, K., Darrell, T.: Efficient Image Matching with Distributions of Local Invariant Features. In: CVPR. (2005)
26. Huttenlocher, D., Klanderman, G., Rucklidge, W.: Comparing images using the Hausdorff distance. PAMI **15**(9) (1993) 850–863
27. Plamondon, R., Srihari, S.: Online and off-line handwriting recognition: a comprehensive survey. PAMI **22** (2000) 63–84
28. Zhang, C., Viola, P.: Multiple-instance pruning for learning efficient cascade detectors. In: NIPS. (2007)
29. Sabzmeydani, P., Mori, G.: Det. peds. by learning shapelet ftrs. In: CVPR. (2007)