DOI:10.1145/2494532

Visual Object Detection with Deformable Part Models

By Pedro Felzenszwalb, Ross Girshick, David McAllester, and Deva Ramanan

Abstract

We describe a state-of-the-art system for finding objects in cluttered images. Our system is based on deformable models that represent objects using local part templates and geometric constraints on the locations of parts. We reduce object detection to classification with latent variables. The latent variables introduce invariances that make it possible to detect objects with highly variable appearance. We use a generalization of support vector machines to incorporate latent information during training. This has led to a general framework for discriminative training of classifiers with latent variables. Discriminative training benefits from large training datasets. In practice we use an iterative algorithm that alternates between estimating latent values for positive examples and solving a large convex optimization problem. Practical optimization of this large convex problem can be done using active set techniques for adaptive subsampling of the training data.

1. INTRODUCTION

Object recognition is a fundamental challenge in computer vision. Consider the problem of detecting objects from a category, such as people or cars, in static images. This is a difficult problem because objects in each category can vary greatly in appearance. Variations arise from changes in illumination, viewpoint, and intra-class variability of shape and other visual properties among object instances. For example, people wear different clothes and take a variety of poses while cars come in various shapes and colors.

Early approaches to object recognition focused on three-dimensional geometric models and invariant features.[22, 24, 25] More contemporary methods tend to be based on appearance-based representations that directly model local image features.[21, 27] Machine learning techniques have been very successful in training appearance-based models in restricted settings such as face detection[29, 30] and handwritten digit recognition.[23] Our system uses new machine learning techniques to train models that combine local appearance models with geometric constraints.

To apply machine learning techniques to object detection we can reduce the problem to binary classification. Consider a classifier that takes as input an image and a position and scale within the image. The classifier determines whether or not there is an instance of the target category at the given position and scale. Detection is performed by evaluating the classifier at a dense set of positions and scales within an image. This approach is commonly called "sliding window" detection. Let x specify an image and a position and scale within the image. In the case of a linear classifier we threshold a score $\beta \cdot \Phi(x)$, where β is a vector of model parameters (often seen as a template) and $\Phi(x)$ is a feature vector summarizing the appearance of an image region defined by x. A difficulty with this approach is that a linear classifier is likely to be insufficient to model objects that can have significant appearance variation.

One representation, designed to handle greater variability in object appearance is that of a pictorial structure,[14, 18] where objects are described by a collection of parts arranged in a deformable configuration. In a pictorial structure model each part encodes local appearance properties of an object, and the deformable configuration is characterized by spring-like connections between certain pairs of parts.

Deformable models such as pictorial structures can capture significant variations in appearance but a single deformable model still cannot represent many interesting object categories. Consider modeling the appearance of bicycles. Bicycles come in different types (e.g., mountain bikes, tandems, penny-farthings with one big wheel and one small wheel) and we can view them from different directions (e.g., frontal versus side views). We use mixtures of deformable models to deal with these more significant variations.

Our classifiers treat mixture component choice and part locations as latent variables. Let x denote an image and a position and scale within the image. Our classifiers compute a score of the form

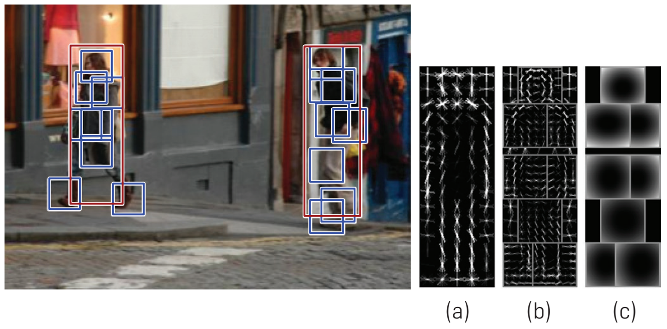
$$f_\beta(x) = \max_z \beta \cdot \Phi(x, z). \qquad (1)$$

Here β is a vector of model parameters, z are latent values, and $\Phi(x, z)$ is a feature vector. If the score is above a threshold, our model will produce a detection at x. Associated with every detection are the inferred latent values, $z^* = \text{argmax}_z\, \beta \cdot \Phi(x, z)$, which specify a mixture component choice and the locations of the parts associated with that component. Figure 1 shows two detections and the inferred part locations in each case. We note that (1) can handle very general forms of latent information. For example, z could specify a derivation under a rich visual grammar.[15]

One challenge in training deformable part models is that it is often difficult to obtain training data with part annotations. Annotating parts can be time consuming and genuinely ambiguous. For example, what are the right parts for a sofa model? We train our models from weakly labeled data in the form of bounding boxes around target objects. Part structure and latent part locations are automatically

This work first appeared in the *IEEE CVPR 2008 Conference* and in the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, No. 9, September 2010.

Figure 1. Detections obtained with a single component person model. The model is defined by a coarse root filter (a), several higher resolution part filters (b), and a spatial model for the location of each part relative to the root (c). The filters specify weights for histogram of oriented gradients features. Their visualization shows the positive weights at different orientations. The visualization of the spatial models reflects the "cost" of placing the center of a part at different locations relative to the root.



inferred during learning. To achieve this, we developed a general framework for discriminative training of latent-variable classifiers of the form in (1). This leads to a formalism that we call latent support vector machine (LSVM).

Sliding window detection leads to imbalanced classification problems. There are vastly more negative examples than positive ones. To obtain high performance using discriminative training it is often important to make exhaustive use of large training sets. This motivates a data subsampling process that searches through all of the negative instances to find the hard negative examples and then trains a model relative to those instances. A heuristic methodology of data mining for hard negatives was adopted by Dalal and Triggs[7] and goes back at least to the training methods used by Schneiderman and Kanade[28] and Viola and Jones.[30] We developed simple data mining algorithms for subsampling the training data for SVMs and LSVMs that are guaranteed to converge to the optimal model defined in terms of the entire training set.

We formally define our models in Section 2. We describe a general framework for learning classifiers with latent variables in Section 3. Section 4 describes how we use this framework to train object detection models. We present experimental results in Section 5 and conclude by discussing related work in Section 6.

2. MODELS

A core component of our models is templates, or filters, that capture the appearance of object parts based on local image features. Filters define scores for placing parts at different image positions and scales. These scores are combined using a deformation model that scores an arrangements of parts based on geometric relationships. Detection involves searching over arrangements of parts using efficient algorithms. This is done separately for each component in a mixture of deformable part models.

2.1. Filters

Our models are built from linear filters that are applied to dense feature maps. A feature map is an array whose entries are d-dimensional feature vectors computed on a dense grid of image locations (e.g., every 8×8 pixels). Each feature vector describes a small image patch while introducing some invariants. The framework described here is independent of the specific choice of features. In practice we use a low-dimensional variation of the histogram of oriented gradient (HOG) features from Dalal and Triggs.[7] HOG features introduce invariances to photometric transformations and small image deformations.

A linear filter is defined by a $w \times h$ array of d-dimensional weight vector. Intuitively, a filter is a template that is tuned to respond to an iconic arrangement of image features. Filters are typically much smaller than feature maps and can be applied at different locations within a feature map. The score, or response, of a filter F at a particular feature map location is obtained by taking the dot product of F's array of weight vectors, concatenated into a single long vector, with the concatenation of the feature vectors extracted from a $w \times h$ window of the feature map. Because objects appear at a wide range of scales, we apply the same filter to multiple feature maps, each computed from a rescaled version of the original image. Figure 2 shows some examples of filters, feature maps, and filter responses. To fix notation, let I be an image and $p = (x, y, s)$ specify a position and scale in the image. We write $F \cdot \phi(I, p)$ for the score obtained by applying filter F at the position and scale specified by p.

2.2. Deformable part models

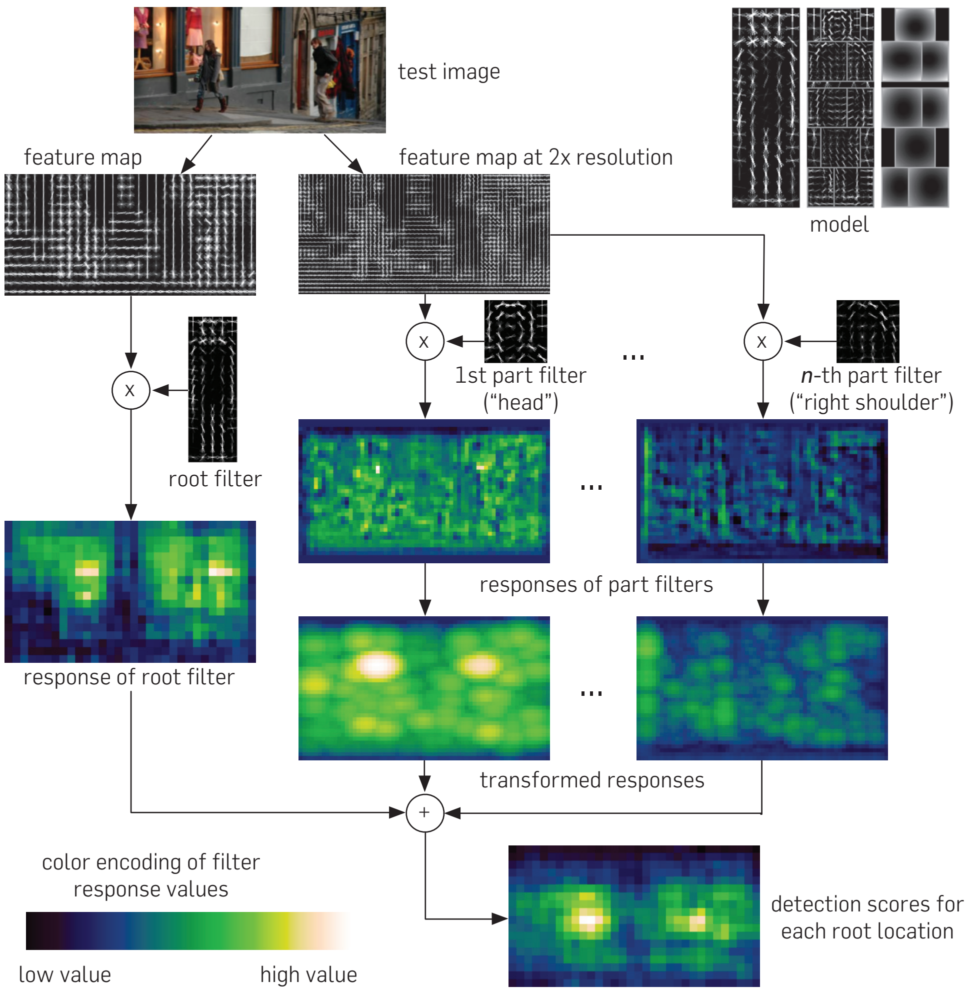
To combine a set of filters into a deformable model we define spring-like connections between some pairs of filters. Thinking of filters as vertices and their pairwise connections as edges, a model is defined by a graph. Here we consider models represented by star graphs, where one filter acts as the hub, or root, to which all other filters are connected.

In our star models, a low resolution root filter, that approximately covers an entire object, serves as the star's hub. Higher resolution part filters, that cover smaller regions of the object, are connected to the root. Figure 1 illustrates a star model for detecting pedestrians and its two highest scoring detections in a test image.

We have found that using higher resolution features for defining part filters is essential for obtaining high recognition performance. With this approach the part filters capture finer resolution features that are localized to greater accuracy when compared to the features captured by the root filter. Consider building a model for a face. The root filter might capture a coarse appearance model for the face as a whole while the part filters might capture the detailed appearance of face parts such as eyes, nose, and mouth.

The model for an object with n parts is defined by a set of parameters $(F_0, (F_1, d_1), \ldots, (F_n, d_n), b)$ where F_0 is a root filter, F_i is a part filter, d_i is a vector of deformation parameters, and b is a scalar bias term. The vector d_i specifies the coefficients of a quadratic function that scores a position for filter i relative to the root filter's position. We use a quadratic deformation model because it is relatively flexible while still amenable to efficient computations. A quadratic score over relative positions can be thought of as a spring that connects a part filter to the root filter. The rest position and rigidity of the spring are determined by d_i.

Figure 2. ***Detection at one scale.*** **Responses from the root and part filters are computed on different resolution feature maps. Distance transforms are used to solve equation (7) efficiently for all possible part placements. The transformed responses are combined to yield a final score for each root location. We show the responses and transformed responses for the "head" and "right shoulder" parts. Note how the "head" filter is more discriminative. The combined scores clearly show two good hypotheses for the object at this scale.**



test image
feature map
feature map at 2x resolution
model
root filter
1st part filter ("head")
n-th part filter ("right shoulder")
responses of part filters
response of root filter
transformed responses
color encoding of filter response values
low value
high value
detection scores for each root location

An object hypothesis is given by a configuration vector $z = (p_0, ..., p_n)$, where $p_i = (x_i, y_i, s_i)$ specifies the position and scale of the i-th filter. The score of a hypothesis is given by the scores of each filter at their respective locations (the data term) minus a deformation cost that depends on the relative position of each part with respect to the root (the spatial prior), plus the bias,

$$\text{score}(z) = \sum_{i=0}^{n} F_i \cdot \phi(I, p_i) - \sum_{i=1}^{n} d_i \cdot \psi(p_i, p_0) + b, \quad (2)$$

$$\text{where} \quad \psi(p_i, p_0) = (dx_i, dy_i, dx_i^2, dy_i^2),$$

$$\text{with} \quad dx_i = x_i - x_0 \quad \text{and} \quad dy_i = y_i - y_0.$$

Each term in the second summation in (2) can be interpreted as a spring deformation model that anchors part i to some ideal location relative to the root.

The score of a hypothesis z can be expressed in terms of a dot product, $\beta \cdot \Phi(I, z)$, between a vector of model parameters β and a feature vector $\Phi(I, z)$,

$$\beta = (F_0, ..., F_n, d_1, ..., d_n, b). \quad (3)$$

$$\Phi(I, z) = (\phi(I, p_0), ... \phi(I, p_n), -\psi(dx_1, dy_1, ..., -\psi(dx_n, dy_n), 1). \quad (4)$$

This makes a connection between deformable part models and linear classifiers. We use this representation for learning the model parameters with the latent SVM framework.

2.3. Detection

To detect objects in an image we compute an accumulated score for each root filter location p_0 according to the best possible placement of the parts relative to p_0

$$\text{score}(p_0) = \max_{p_1, ..., p_n} \text{score}(p_0, ..., p_n) \quad (5)$$

$$= F_0 \cdot \phi(I, p_0) + \sum_{i=1}^{n} m_i(p_0) + b \quad (6)$$

$$\text{where} \quad m_i(p_0) = \max_{p_i} F_i \cdot \phi(I, p_i) - d_i \cdot \psi(p_i, p_0). \qquad (7)$$

Let k be the number of possible locations for each filter. A naive computation of the accumulated score for p_0 would take $O(nk)$ time. Since there are k choices for p_0 this would lead to an $O(nk^2)$ time algorithm for computing all accumulated scores. A much faster approach can be obtained using the generalized distance transform algorithms from Felzenszwalb and Huttenlocher.[13] This leads to a method that computes all of the accumulated scores in $O(nk)$ time total. The approach is illustrated in Figure 2.

We obtain a set of detections by finding the local maxima of score(p_0) that exceed a user-specified confidence threshold. This non-maximal suppression step removes redundant detections that differ slightly in position and scale and thus are largely supported by the same image evidence.

2.4. Mixture models

As described in the introduction many interesting object categories exhibit more intra-class variation than can be accounted for by a single deformable model. A natural extension involves using a mixture of deformable models.

Formally, a mixture model with m components is defined by a m-tuple, $M = (M_1, \ldots, M_m)$, where M_c is the model for the c-th component. An object hypothesis, $z = (c, p_0, \ldots, p_{n_c})$ for a mixture model specifies a mixture component, $1 \leq c \leq m$, and a location p_i for each filter of M_c. The score of this hypothesis is the score of the hypothesis $z' = (p_0, \ldots, p_{n_c})$ for the c-th model component. As in the case of a single deformable model the score of a hypothesis for a mixture model can be expressed by a dot product between a vector of model parameters (the concatenation of the parameters for each mixture component) and an appropriately constructed feature vector that depends on the image I and the hypothesis z.

To detect objects using a mixture model, we first compute the accumulated root scores independently for each component, and then for each root location we select the highest scoring component hypothesis at that location.

3. LATENT SVM

Our models involve binary classifiers with latent variables. To train these classifiers we use a latent support vector machine (LSVM).[a] To formulate the LSVM training objective consider scoring functions of the following form

$$f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z). \qquad (8)$$

Here x is an input, such as a detection window; β is a vector of model parameters; and z is an assignment of values to latent variables such as part placements. The set $Z(x)$ defines the possible latent values for an example x. A binary label for x can be obtained by thresholding this score.

In analogy to classical SVMs we can train β from labeled examples $D = (\langle x_1, y_1 \rangle, \ldots, \langle x_n, y_n \rangle)$, where $y_i \in \{-1, 1\}$, by minimizing the following objective function,

$$L_D(\beta) = \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{n} \max\left(0, 1 - y_i f_\beta(x_i)\right), \qquad (9)$$

where $\max(0, 1 - y_i f_\beta(x_i))$ is the standard hinge loss and the constant C controls the relative weight of the regularization term. Note that if there is a single possible latent value for each example ($|Z(x_i)| = 1$) then f_β is linear in β, and we obtain linear SVMs as a special case of LSVMs.

3.1. Semi-convexity

Because the scoring function (8) is nonlinear in β, the LSVM objective function (9) is non-convex in β. However, the training problem becomes convex once latent information is specified for the positive training examples.

To see this, note that $f_\beta(x)$ as defined in (8) is a maximum of functions each of which is linear in β. Hence $f_\beta(x)$ is a max of convex functions and is hence convex. This implies that the hinge loss, $\max(0, 1 - y_i f_\beta(x_i))$, is convex in β when $y_i = -1$. That is, the loss function is convex in β for negative examples. Now if we only allow a single setting of the latent variables for each positive example, i.e., if we fix the latent values for the positives, then the hinge loss becomes linear in β, and hence convex, on the positive examples also. So fixing the latent information on the positive examples makes the overall training objective convex. This observation motivates the following training algorithm:

1. Holding β fixed, select the best latent value for each positive example,
$$z_i = \underset{z \in Z(x_i)}{\operatorname{argmax}} \, \beta \cdot \Phi(x, z).$$
2. Fixing the latent variables for the positive examples to $Z(x_i) = \{z_i\}$, solve the (now convex) optimization problem defined by (9).

This procedure can be seen as a block coordinate descent optimization of an auxiliary training objective $L(\beta, Z_p)$ that depends on both β and a choice of latent values for the positive examples Z_p. Moreover, if the pair (β, Z_p) minimizes the auxiliary objective $L(\beta, Z_p)$ then β minimizes the original LSVM objective $L(\beta)$. This justifies training via minimization of $L(\beta, Z_p)$. The semi-convexity property plays an important role in this approach because it leads to a convex optimization problem in Step 2, even though the latent values for the negative examples are not fixed.

3.2. Optimization with data subsampling

When the latent values for the positive examples are fixed the LSVM objective function is convex and can be optimized using a variety of methods. However, a classical difficulty that arises when training a sliding window classifier is that a single training image yields an overwhelming number of negative examples. This difficulty has been previously addressed using heuristics that start with a small subset of the negative examples and alternate between training a model and growing the negative training set with false positives generated by the previous model.[7, 30]

We have developed a version of this heuristic process that is tailored for discriminative training with a hinge loss. It involves repeatedly training models using relatively small

[a] A latent SVM is equivalent to a multiple instance SVM.[2]

subsets of the training data and is guaranteed to find an optimal model under the original large dataset. The approach is applicable for both standard SVM and latent SVM.

Our method maintains a subset C of the training data, trains the model parameters β on the subset C, and then updates C. To describe the procedure more formally, we first define the "hard examples" for a model β as follows, where i ranges over the full training set

$$H(\beta) = \{(x_i, y_i) : y_i f_\beta(x_i) \leq 1\}.$$

Our algorithm initializes C to an arbitrary set of examples (such as all positives and a small random subset of negatives). It then repeats the following steps where $\beta^*(C)$ is the model minimizing the training objective on the training set C.

1. Set $\beta := \beta^*(C)$.
2. Shrink C by removing elements not in $H(\beta)$.
3. Grow C by adding new examples from $H(\beta)$.

Recall that we are holding the latent values of positive examples fixed, so the objective function is convex. If C contains all of $H(\beta)$ after Step 1 then the subgradients of the training objective with respect to C equal the subgradients of the training objective with respect to the entire dataset and we can terminate the process. Furthermore, we can prove that the process will always terminate. The basic insight is that the value of the training objective on the set C is non-decreasing. Note that the training objective on C does not change when we shrink C in Step 2, because the hinge loss of the examples being removed is zero. The objective also does not decrease when new elements are added to C in Step 3. In fact the training objective on C grows over time and since the number of possible subsets C is finite, the process must terminate.

4. TRAINING MODELS

Suppose we have training images with bounding boxes around objects in a particular category. We define a positive example from each bounding box. Bounding boxes do not specify mixture component labels or filter locations, so we treat these as latent variables during training. We use the bounding box information to constraint the placement of the root filter in each positive example. We define a very large set of negative examples from images that do not contain objects from the target category. Each position and scale in such an image yields a different negative example.

Together, the positive and negative examples lead to a latent SVM training problem where we want to select a model that gives high score for positive examples and low score for negatives. We use the block coordinate descent algorithm from Section 3.1 to optimize the LSVM training objective. Since this algorithm is susceptible to local minima it must be initialized carefully.

Initialization: We begin by learning root filters for each component of a mixture model. We partition the positive examples into m disjoint groups based on the aspect ratio of their bounding boxes. For each group, we warp the image data in the bounding boxes to a canonical size and train a root filter with a standard SVM. To initialize the part filters, we greedily place a fixed number of parts (eight, in all of our experiments) to cover high-energy regions of the root filter. The part filter coefficients are initialized by interpolating the root filter to twice the spatial resolution, and the part deformation parameters are initialized to a value that penalizes large displacements. Figure 3 shows the initial model obtained for a two-component car model.

Coordinate descent: Given an initial model β, Step 1 of the coordinate descent algorithm estimates latent values for each positive example. This includes a mixture component label and filter locations. We constrain the locations of the root filters to placements that overlap with the bounding box of a positive example by a significant amount. During Step 2 of coordinate descent, we learn a new model β by solving a large scale convex program with stochastic subgradient descent and data subsampling over the negative examples (Section 3.2). Note that we repeatedly update the latent values for each positive example, including a mixture component label. Therefore our algorithm naturally performs a "discriminative clustering" of the positive examples.

5. EMPIRICAL RESULTS

The system described here has been evaluated on the PASCAL VOC datasets. We refer to Everingham et al.[9] for details, but emphasize that the PASCAL VOC challenges are widely acknowledged as difficult testbeds for object detection.

Each dataset contains thousands of real-world images, and specifies ground-truth bounding boxes for 20 object classes. At test time, the goal is to predict the bounding boxes of all objects of a given class in an image (if any). In practice a system will output a set of bounding boxes with confidence scores, and these scores are thresholded at different points to obtain a precision-recall curve across all images in the test set. For a particular threshold the precision is the fraction of the reported bounding boxes that are correct detections, while recall is the fraction of the objects found.

A reported bounding box is considered correct if it overlaps more than 50% with a ground-truth bounding box. When a system reports several bounding boxes that overlap with a single ground-truth bounding box, only one detection is considered correct. One scores a system by the average precision (AP) of its precision-recall curve, computed for each object class independently.

Figure 3. Initialization. (a) The initial root filters for a car model. (b) and (c) The part filters and deformation models initialized from (a).

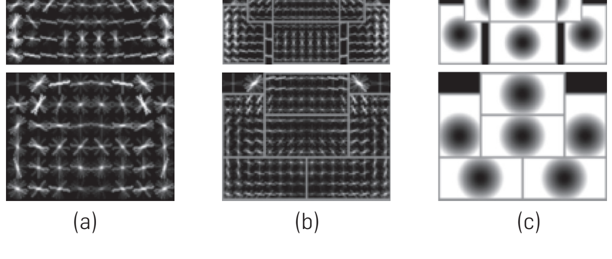


Figure 4. Visualizations of some of the models learned on the PASCAL 2010 dataset.

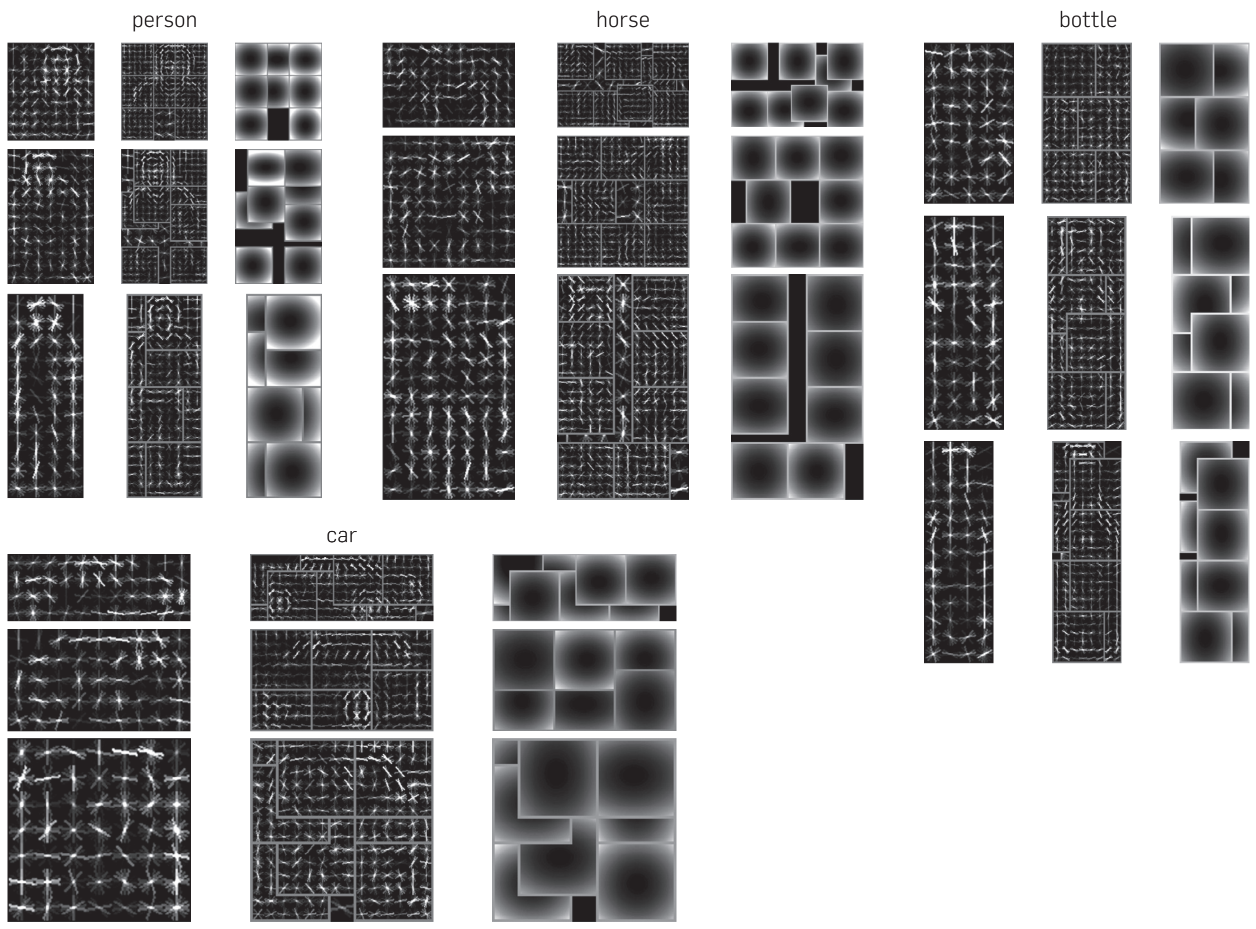


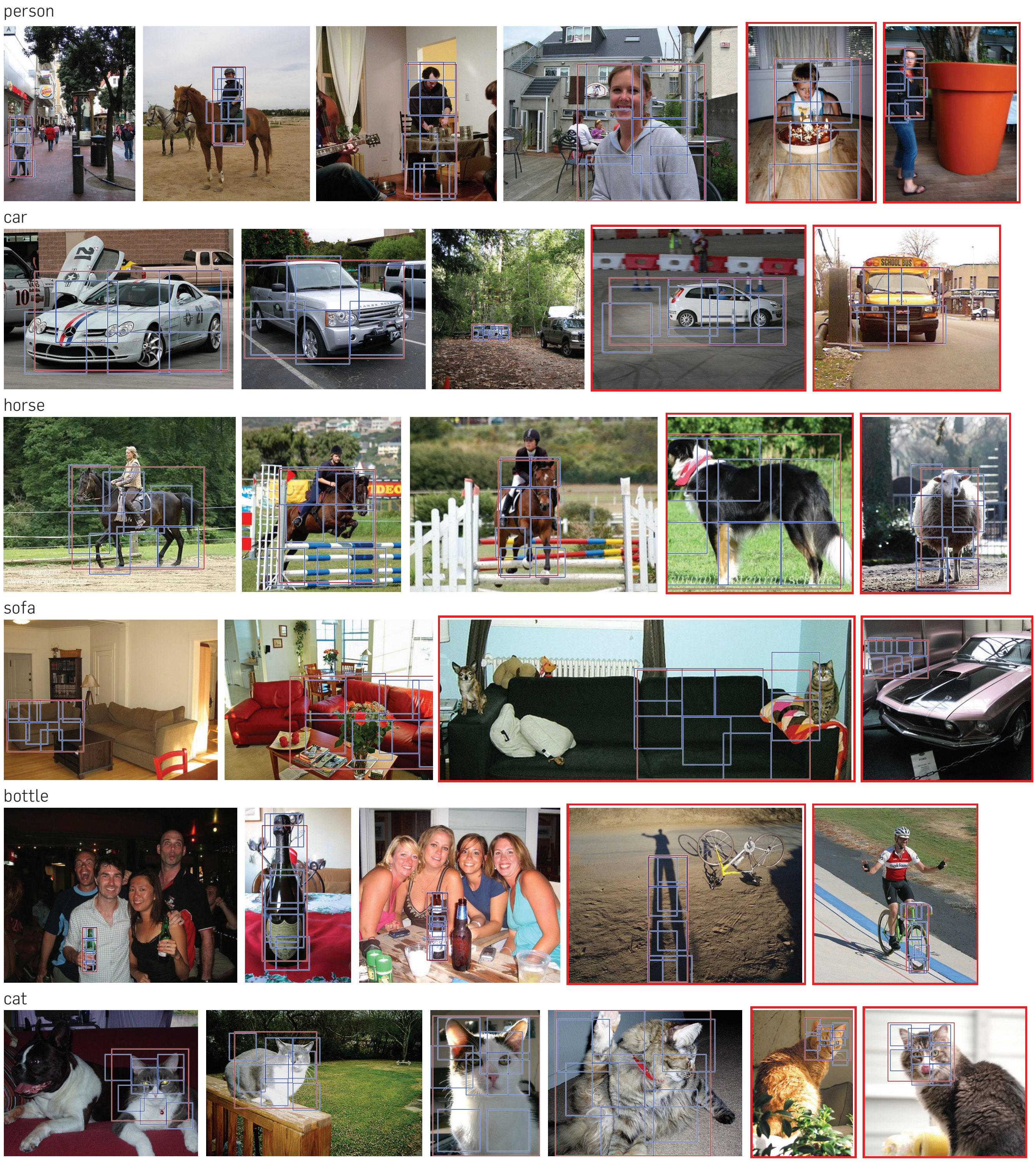
Figure 4 shows some models learned from the PASCAL VOC 2010 dataset. Figure 5 shows some example detections using those models. We show both high-scoring correct detections and high-scoring false positives. These examples illustrate how our models can handle significant variations in appearance such as in the case of cars and horses.

In some categories our false detections are often due to similarities among objects in different categories, such as between horse and cow or between car and bus. In other categories false detections are often due to the relatively strict bounding box overlap criteria. The two false positives shown for the person category are due to insufficient overlap with the ground-truth bounding box. The same is true for the cat category, where we often detect the face of a cat and report a bounding box that has relatively little overlap with the correct bounding box that encloses the whole object. In fact, the top 20 highest scoring false positive detections for the cat category correspond to a cat face. This is an extreme case but it gives an explanation for our low AP score in this category. Many positive training examples of cats contain only the face, and our cat mixture model has a component dedicated to detect cat faces, while another component captures an entire cat. Sometimes the wrong mixture component has the highest score, suggesting that our scores across different components could be better calibrated.

In the 2007, 2008, and 2009 PASCAL VOC competitions our system obtained the highest AP score in 6, 7, and 7 out of 20 categories, respectively.[9] Our entry was declared the winner of the competition in 2008 and 2009. In the 2010 competition, our system won in 3 of 20 categories, and the 3 systems that achieved a higher mean AP score (averaged over all classes) were all extensions of our system using additional features, richer context, and more parts.[9] Table 1 summarizes the AP scores of our system on the 2010 dataset, together with the best scores achieved across all systems that entered the official competition. We also show the effect of two post-processing methods that improve the quality of our detections.

The first method, bounding-box prediction, demonstrates the added benefit that comes with inferring latent structure at test time. We use a linear regression model to predict the true bounding box of a hypothesis from the inferred part configuration. The second method, context rescoring, computes a new confidence score for each detection with a polynomial kernel SVM whose features are the base detection score and the highest score for each of the 20 object-class detectors in the same image. This method can learn co-occurrence constraints between object classes; because cars and sofas tend not to co-occur, car detections should be downweighted if an image contains a high-scoring sofa. This context rescoring method

Figure 5. Examples of high-scoring detections on the PASCAL 2007 dataset. The red-framed images (last two in each row) illustrate false positives for each category. Many false positives (such as for person and cat) are due to the stringent bounding box overlap criteria.



currently outperforms more complex approaches, such as that proposed by Desai et al.[8]

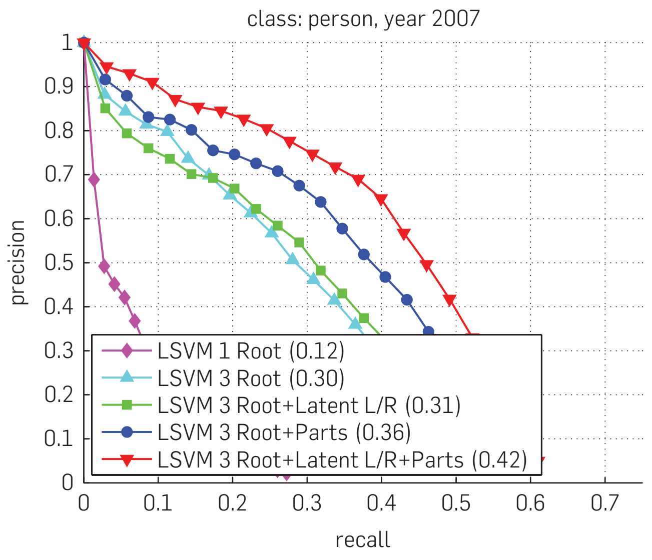
We evaluated different aspects of our system on the longer-established PASCAL VOC 2007 dataset. Figure 6 summarizes results of different models for the person category. We trained models with 1 and 3 components, with and without parts, and forcing mirror symmetry in each component or allowing for asymmetric models. We see that the use of parts can significantly improve the detection accuracy. Mixture models are also very important in the

Table 1. ***PASCAL VOC 2010 results.***

	Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	Mbike	Person	Plant	Sheep	Sofa	Train	TV
Base[a]	47.2	50.8	8.6	12.2	32.2	48.9	44.4	28.1	13.6	22.7	11.3	17.4	40.4	47.7	44.4	7.6	30.0	17.3	38.5	34.3
BB[b]	48.7	52.0	8.9	12.9	32.9	51.4	47.1	29.0	13.8	23.0	11.1	17.6	42.1	49.3	45.2	7.4	30.8	17.1	40.6	35.1
Context[c]	52.4	54.3	13.0	15.9	35.1	54.2	49.1	31.8	15.5	26.2	13.5	21.5	45.4	51.6	47.5	9.1	35.1	19.4	46.6	38.0
Best[d]	58.4	55.3	19.2	21.0	**35.1**	55.5	**49.1**	47.7	20.0	31.5	27.7	37.2	51.9	56.3	**47.5**	13.0	37.8	33.0	50.3	41.9

[a]Score of our base system.
[b]The system with bounding box prediction.
[c]The final system with context rescoring.
[d]The highest score over all systems entered into the 2010 competition (bolded numbers indicate that our system obtained the highest score).

Figure 6. ***Precision/Recall curves for models trained on the person category of the PASCAL 2007 dataset.*** **We show results for 1- and 3-component models with and without parts. For the 3-component models, we show results where the models are forced to be symmetric and where the models are allowed to be asymmetric and left vs. right orientation is treated as a latent variable during both training and testing ("Latent L/R"). In parentheses we show the average precision score for each model.**



person category where there are many examples of people truncated at various heights (e.g., by desks). Allowing for asymmetric models, where the object's facing direction is treated as a latent variable, produces a very small change when working with root-filter only models. However, after adding parts, latent direction yields a significant improvement.

6. DISCUSSION

Object detection is difficult because instances can vary greatly in appearance and because objects tend to appear in cluttered backgrounds. Latent-variable models provide a natural formalism for dealing with appearance variation. This represents a departure from other approaches that rely primarily on invariant features.[26] Rather, we find that a combination of both approaches, namely a latent variable model built on top of an invariant local image descriptor,[7] works quite well. Our models are robust to cluttered backgrounds by way of their discriminative training. This requires using a very large number of negative training examples to emulate the distribution of positive and negatives encountered at test-time.

There is a rich body of work in the use of deformable models of various types for object detection, including several kinds of deformable template models (e.g., Cootes et al.,[4] Coughlan et al.,[5] Grenander et al.,[20] and Yuille et al.[33]) and a variety of part-based models (e.g., Amit and Trouve,[1] Burl et al.,[3] Crandall et al.,[6] Felzenszwalb and Huttenlocher,[14] Fergus et al.,[17] Fischler and Elschlager,[18] and Weber et al.[31]). Our models are based on the pictorial structures formulation from Felzenszwalb and Huttenlocher[14] and Fischler and Elschlager[18], which evaluates a dense set of possible part positions and scales in an image. We are able to do so in an efficient manner using the fast matching algorithms of Felzenszwalb and Huttenlocher.[14] Our approach differs from past work on deformable models with its use of highly engineered local features[7] and weakly supervised discriminative learning algorithms.

The work described here was originally published in Felzenszwalb et al.[12] and Felzenszwalb et al.[16] with associated code releases available online.[11] We have extended this work in a variety of ways. In Felzenszwalb et al.[10] we explored cascade classifiers that evaluate the filters in a deformable part model sequentially and prune the computation using intermediate thresholds. This approach results in an order-of-magnitude speedup and real-time performance with little loss in accuracy. In Felzenszwalb and McAllester[15] and Girshick et al.,[19] we pursued grammar-based models that generalized deformable part models to allow for objects with variable structure, mixture models at the part level and reusability of parts across components and object classes. Finally, our method is still limited somewhat by its sensitivity to initialization. One approach to reducing this sensitivity is to use partially or fully annotated data with part and mixture labels. Our recent work has shown that one can use such a framework to achieve competitive results for facial analysis[34] and articulated pose estimation.[32]

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. IIS 0746569, IIS 0811340 and IIS 0812428.

References

1. Amit, Y., Trouve, A. POP: Patchwork of parts models for object recognition. *Int. J. Comput. Vis. 75*, 2 (2007), 267–282.
2. Andrews, S., Tsochantaridis, I., Hofmann, T. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems* (2003), volume 15.
3. Burl, M., Weber, M., Perona, P. A probabilistic approach to object recognition using local photometry and global geometry. In *European Conference on Computer Vision* (1998).
4. Cootes, T., Edwards, G., Taylor, C. Active appearance models. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 6 (2001), 681–685.
5. Coughlan, J., Yuille, A., English, C., Snow, D. Efficient deformable template detection and localization without user initialization. *Comput. Vis. Image Understand. 78*, 3 (2000), 303–319.
6. Crandall, D., Felzenszwalb, P., Huttenlocher, D. Spatial priors for part-based recognition using statistical models. In *IEEE Conference on Computer Vision and Pattern Recognition* (2005).
7. Dalal, N., Triggs, B. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition* (2005).
8. Desai, C., Ramanan, D., Fowlkes, C. Discriminative models of multi-class object layout. *Int. J. Comput. Vis. 95*, 1 (2011), 1–12.
9. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A. The PASCAL Visual Object Classes Challenges. http://www.pascal-network.org/challenges/VOC/index.html.
10. Felzenszwalb, P., Girshick, R., McAllester, D. Cascade object detection with deformable part models. In *IEEE Computer Vision and Pattern Recognition* (2010).
11. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D. Discriminatively trained deformable part models. http://people.cs.uchicago.edu/~pff/latent/.
12. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D. Object detection with discriminatively trained part based models. *IEEE Trans. Pattern Anal. Mach. Intell. 32*, 9 (2010), 1627–1645.
13. Felzenszwalb, P., Huttenlocher, D. Distance transforms of sampled functions. Technical Report 2004–1963, CIS Dept., Cornell University, 2004.
14. Felzenszwalb, P., Huttenlocher, D. Pictorial structures for object recognition. *Int. J. Comput. Vis. 61*, 1 (2005), 55–79.
15. Felzenszwalb, P., McAllester, D. Object detection grammars. Technical Report TR-2010–02, CS Dept., University of Chicago, 2010.
16. Felzenszwalb, P., McAllester, D., Ramanan, D. A discriminatively trained, multiscale, deformable part model. In *IEEE Conference on Computer Vision and Pattern Recognition* (2008).
17. Fergus, R., Perona, P., Zisserman, A. Object class recognition by unsupervised scale-invariant learning. In *IEEE Conference on Computer Vision and Pattern Recognition* (2003).
18. Fischler, M., Elschlager, R. The representation and matching of pictorial structures. *IEEE Trans. Comput. C-22*, 1 (1973), 67–92.
19. Girshick, R., Felzenszwalb, P., McAllester, D. Object detection with grammar models. In *Advances in Neural Information Processing Systems* (2011), volume 24.
20. Grenander, U., Chow, Y., Keenan, D. *HANDS: A Pattern-Theoretic Study of Biological Shapes*, Springer-Verlag, 1991.
21. Huttenlocher, D., Klanderman, G., Rucklidge, W. Comparing images using the hausdorff distance. *IEEE Trans. Pattern Anal. Mach. Intell. 15*, 9 (1993), 850–863.
22. Lamdan, Y. Wolfson, H. Geometric hashing: A general and efficient model-based recognition scheme. In *IEEE International Conference on Computer Vision* (1988).
23. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE 86*, 11 (1998), 2278–2324.
24. Lowe, D. Three-dimensional object recognition from single two-dimensional images. *Artif. intell. 31*, 3 (1987), 355–395.
25. Marr, D., Nishihara, H. Representation and recognition of the spatial organization of three-dimensional shapes. *Proc. Roy. Soc. Lond. B Biol. Sci. 200*, 1140 (1978), 269–294.
26. Mundy, J., Zisserman, A., et al. *Geometric Invariance in Computer Vision*, volume 92, MIT press, Cambridge, MA, 1992.
27. Murase, H., Nayar, S. Visual learning and recognition of 3-d objects from appearance. *Int. J. Comput. Vis. 14*, 1 (1995), 5–24.
28. Schneiderman, H., Kanade, T. A statistical method for 3D object detection applied to faces and cars. In *IEEE Conference on Computer Vision and Pattern Recognition* (2000).
29. Sung, K.K., Poggio, T. Example-based learning for view-based human face detection. *IEEE Trans. Pattern Anal. Mach. Intell. 20*, 1 (1998), 39–51.
30. Viola, P., Jones, M. Robust real-time face detection. *Int. J. Comput. Vis. 57*, 2 (2004), 137–154.
31. Weber, M., Welling, M., Perona, P. Towards automatic discovery of object categories. In *IEEE Conference on Computer Vision and Pattern Recognition* (2000).
32. Yang, Y., Ramanan, D. Articulated pose estimation using flexible mixtures of parts. In *IEEE Conference on Computer Vision and Pattern Recognition* (2011).
33. Yuille, A., Hallinan, P., Cohen, D. Feature extraction from faces using deformable templates. *Int. J. Comput. Vis. 8*, 2 (1992), 99–111.
34. Zhu, X., Ramanan, D. Face detection, pose estimation, and landmark localization in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition* (2012).

Pedro Felzenszwalb School of Engineering and Department of Computer Science, Brown University.

Ross Girshick EECS, UC Berkeley.

David McAllester Toyota Technological Institute at Chicago.

Deva Ramanan Department of Computer Science, UC Irvine.

University of Colorado
Denver

ASSISTANT PROFESSOR
Computer Science and Engineering

The Department of Computer Science and Engineering in the College of Engineering and Applied Science at the University of Colorado Denver invites applications for the position of Assistant Professor.

The candidate will be expected to develop and teach lecture and laboratory courses at all levels, establish an active, externally funded research program; conduct high quality research involving students at all levels, leading to sponsored research and refereed publications; advise students; and contribute to Department, College, and University governance and to the profession. The candidate must have a Ph.D. in Computer Science or closely related field and demonstrated expertise in computer science as evidenced by the candidate's record.

Salary is competitive and commensurate with qualifications and experience. Additional details are available and applications are accepted electronically at **https://www.jobsatcu.com** (refer to job posting #F00552). Screening of applications begins December 1, 2013, and continues until the position is filled. This position is expected to start August 2014. Questions should be directed to **CSE.Search@ucdenver.edu**.

The University of Colorado Denver is dedicated to ensuring a safe and secure environment for our faculty, staff, students and visitors. To assist in achieving that goal, we conduct background investigations for all prospective employees. Some positions may require a motor vehicle report.

The University of Colorado is committed to diversity and equality in education and employment.