Journal of Machine Learning Research 9 (2008) 2549-2578 Submitted 10/07; Published 11/08

Stationary Features and Cat Detection

François Fleuret FLEURET@IDIAP.CH
IDIAP Research Institute,
Centre du Parc, Rue Marconi 19,
Case Postale 592,
1920 Martigny, Switzerland

Donald Geman GEMAN@JHU.EDU
Johns Hopkins University,
Clark Hall 302A,
3400 N. Charles Street
Baltimore, MD 21218, USA

Editor: Pietro Perona

Abstract

Most discriminative techniques for detecting instances from object categories in still images consist of looping over a partition of a pose space with dedicated binary classifiers. The efficiency of this strategy for a complex pose, that is, for fine-grained descriptions, can be assessed by measuring the effect of sample size and pose resolution on accuracy and computation. Two conclusions emerge: (1) fragmenting the training data, which is inevitable in dealing with high in-class variation, severely reduces accuracy; (2) the computational cost at high resolution is prohibitive due to visiting a massive pose partition.

To overcome data-fragmentation we propose a novel framework centered on pose-indexed features which assign a response to a pair consisting of an image and a pose, and are designed to be stationary: the probability distribution of the response is always the same if an object is actually present. Such features allow for efficient, one-shot learning of pose-specific classifiers. To avoid expensive scene processing, we arrange these classifiers in a hierarchy based on nested partitions of the pose; as in previous work on coarse-to-fine search, this allows for efficient processing.

The hierarchy is then ”folded” for training: all the classifiers at each level are derived from one base predictor learned from all the data. The hierarchy is ”unfolded” for testing: parsing a scene amounts to examining increasingly finer object descriptions only when there is sufficient evidence for coarser ones. In this way, the detection results are equivalent to an exhaustive search at high resolution. We illustrate these ideas by detecting and localizing cats in highly cluttered greyscale scenes.

Keywords: supervised learning, computer vision, image interpretation, cats, stationary features, hierarchical search

1. Introduction

This work is about a new strategy for supervised learning designed for detecting and describing instances from semantic object classes in still images. Conventional examples include faces, cars and pedestrians. We want to do more than say whether or not there are objects in the scene; we want to provide a description of the pose of each detected instance, for example the locations of certain landmarks. More generally, pose could refer to any properties of object instantiations which

are not directly observed; however, we shall concentrate on geometric descriptors such as scales, orientations and locations.

The discriminative approach to object detection is to induce classifiers directly from training data without a data model. Generally, one learns a pose-specific binary classifier and applies it many times (Rowley et al., 1998; Papageorgiou and Poggio, 2000; Viola and Jones, 2004; LeCun et al., 2004). Usually, there is an outer loop which visits certain locations and scales with a sliding window, and a purely learning-based module which accommodates all other sources of variation and predicts whether or not a sub-window corresponds to a target. Parsing the scene in this manner already exploits knowledge about transformations which preserve object identities. In particular, translating and scaling the training images to a reference pose allows for learning a base classifier with all the training examples. We refer to such learning methods, which use whole image transforms in order to normalize the pose, as "data-aggregation" strategies.

However such transforms, which must be applied online during scene parsing as well as offline during training, may be costly, or even ill-defined, for complex poses. How does one "normalize" the pose of a cat? In such cases, an alternative strategy, which we call "data-fragmentation," is to reduce variation by learning many separate classifiers, each dedicated to a sub-population of objects with highly constrained poses and each trained with only those samples satisfying the constraints. Unfortunately, this approach to invariance might require a massive amount of training data due to partitioning the data. As a result, the discriminative approach has been applied almost exclusively to learning rather coarse geometric descriptions, such as a facial landmark and in-plane orientation, by some form of data-aggregation. Summarizing: aggregating the data avoids sparse training but at the expense of costly image transforms and restrictions on the pose; fragmenting the data can, in principle, accommodate a complex pose but at the expense of crippling performance due to impoverished training.

A related trade-off is the one between computation and pose resolution. Sample size permitting, a finer subpopulation (i.e., higher pose resolution) allows for training a more discriminating classifier. However, the more refined the pose partitioning, the more online computation because regardless of how the classifiers are trained, having more of them means more costly scene parsing. This trade-off is clearly seen for cascades (Viola and Jones, 2004; Wu et al., 2008): at a high true positive rate, reducing false positives could only come at the expense of considerable computation due to dedicating the cascade to a highly constrained pose, hence increasing dramatically the number of classifiers to train and evaluate in order to parse the scene.

To set the stage for our main contribution, a multi-resolution framework, we attempted to quantify these trade-offs with a single-resolution experiment on cat detection. We considered multiple partitions of the space of poses at different resolutions or granularities. For each partition, we built a binary classifier for each cell. There are two experimental variables besides the resolution of the partition: the data may be either fragmented or aggregated during training and the overall cost of executing all the classifiers may or may not be equalized. Not surprisingly, the best performance occurs with aggregated training at high resolution, but the on-line computational cost is formidable. The experiment is summarized in an Appendix A and described in detail in Fleuret and Geman (2007).

Our framework is designed to avoid these trade-offs. It rests on two core ideas. One, which is not new, is to control online computation by using a hierarchy of classifiers corresponding to a recursive partitioning of the pose space, that is, parameterizations of increasing complexity. A richer parametrization is considered only when "necessary", meaning the object hypothesis cannot

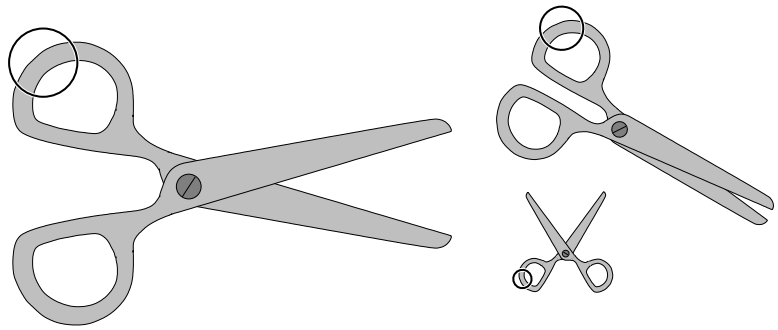


Figure 1: An idealized example of stationary features. The pose of the scissors could be the locations of the screw and the two tips, in which case one might measure the relative frequency a particular edge orientation inside in a disc whose radius and location, as well as the chosen orientation, depends on the pose. If properly designed, the response statistics have a distribution which is invariant to the pose when in fact a pair of scissors is present (see § 3.3).

be ruled out with a simpler one (see, e.g., Fleuret and Geman, 2001; Stenger et al., 2006). (Note that cascades are efficient for a similar reason - they are coarse-to-fine in terms of background rejection.) However, hierarchical organization alone is unsatisfactory because it does not solve the data-fragmentation problem. Unless data can be synthesized to generate many dedicated sets of positive samples, one set per node in the hierarchy, the necessity of training a classifier for every node leads to massive data fragmentation, hence small node-specific training sets, which degrades performance.

The second idea, the new one, is to avoid data-fragmentation by using pose-specific classifiers trained with "stationary features", a generalization of the underlying implicit parametrization of the features by a scale and a location in all the discriminative learning techniques mentioned earlier. Each stationary feature is "pose-indexed" in the sense of assigning a numerical value to each combination of an image and a pose (or subset of poses). The desired form of stationarity is that, for any given pose, the *distribution* of the responses of the features over images containing an object at that pose does not depend on the pose. Said another way, if an image and an object instance at a given pose are selected, and only the responses of the stationary features are provided, one cannot guess the pose. This is illustrated in Figure 1: knowing only the proportion of edges at a pose-dependent orientation in the indicated disk provides no information about the pose of the scissors.

Given that objects are present, a stationary feature evaluated at one pose is then the "same" as at any other, but not in a literal, point-wise sense as functions, but rather in the statistical, population sense described above. In particular, stationary features are not "object invariants" in the deterministic sense of earlier work (Mundy and Zisserman, 1992) aimed at discovering algebraic and geometric image functionals whose actual values were invariant with respect to the object pose. Our aim is less ambitious: our features are only "invariant" in a statistical sense. But this is enough to use all the data to train each classifier.

Of course the general idea of connecting features with object poses is relatively common in object recognition. As we have said, pose-indexing is done implicitly when transforming images to a

reference location or scale, and explicitly when translating and scaling Haar wavelets or edge detectors to compute the response of a classifier for a given location and scale. Surprisingly, however, this has not been formulated and analyzed in general terms, even though stationarity is all that is needed to aggregate data while maintaining the standard properties of a training set. Stationarity makes it possible, and effective, to analytically construct an entire family of pose-specific classifiers—all those at a given level of the hierarchy—using one base classifier induced from the entire training set. In effect, each pose-specific classifier is a "deformation" of the base classifier. Hence the number of classifiers to train grows linearly, not exponentially, with the depth of the pose hierarchy. This is what we call a folded hierarchy of classifiers: a tree-structured hierarchy is collapsed, like a fan, into a single chain for training and then expanded for coarse-to-fine search.

The general formulation opens the way for going beyond translation and scale, for example for training classifiers based on checking consistency among parts or deformations of parts instead of relying exclusively on their marginal appearance. Such a capability is indeed exploited by the detector we designed for finding cats and greatly improves the performance compared to individual part detection. This gain is shown in Figure 2, the main result of the paper, which compares ROC curves for two detectors, referred to as "H+B" and "HB" in the figure. In the "H+B" case, two separate detectors are trained by data aggregation, one dedicated to heads and the other to bodies; the ROC curve is the best we could do in combining the results. The "HB" detector is a coordinated search based on stationary features and a two-level hierarchy; the search for the belly location in the second-level is conditional on a pending head location and data fragmentation is avoided with pose-indexed features in a head-belly frame. A complete explanation appears in § 6.

In §2, we summarize previous, related work on object detection in still images. Our notation and basic ideas are formally introduced in §3, highlighting the difference between transforming the signal and the features. The motivational experiment, in which we substantiate our claims about the forced trade-offs when conventional approaches are applied to estimating a complex pose, could be read at this point; see Appendix A. Embedding pose-indexed classifiers in a hierarchy is described in §4 and the base classifier, a variation on boosting, is described in §5. In §6 we present our main experiment - an application of the entire framework, including the specific base features, pose hierarchy and pose-indexed features, to detecting cats in still images. Finally, some concluding remarks appear in §7.

2. Related Work

We characterize other work in relation to the two basic components of our detection strategy: explicit modeling of a hidden pose parameter, as in many generative and discriminative methods, and formulating detection as a controlled "process of discovery" during which computation is invested in a highly adaptive and unbalanced way depending on the ambiguities in the data.

2.1 Hidden Variables

A principal source of the enormous variation in high-dimensional signals (e.g., natural images) is the existence of a hidden state which influences many components (e.g., pixel intensities) simultaneously, creating complex statistical dependencies among them. Still, even if this hidden state is of high dimension, it far simpler than the observable signal itself. Moreover, since our objective is to interpret the signal at a semantic level, much of the variation in the signal is irrelevant.

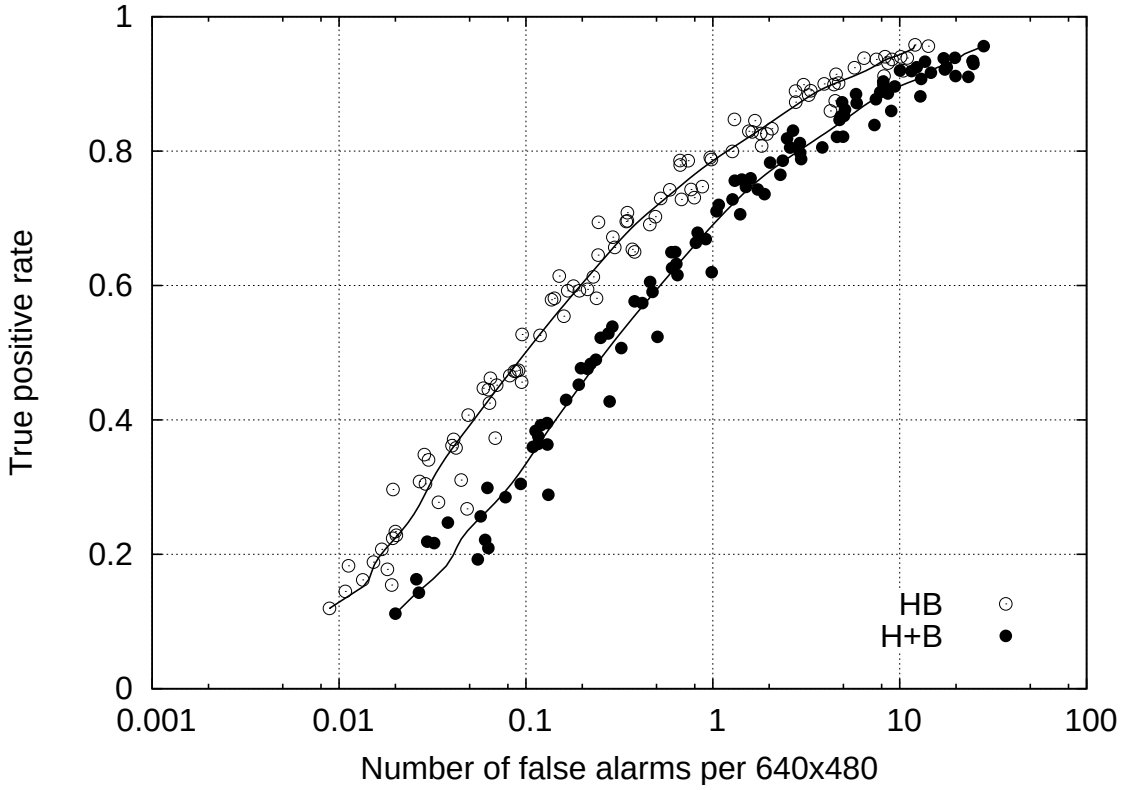


Figure 2: ROC curves for head-belly detection. The criterion for a true detection is that the estimates of the head location, head size and belly location all be close to the true pose (see § 6.6). The H+B detector is built from separate head and body detectors while the HB detector is built upon pose indexed features (see § 6.5).

In fact, conditioning on the value of the hidden state, which means, in practice, testing for the presence of a target with a given pose, often leads to very simple, yet powerful, statistical models by exploiting the increased degree of independence among the components of the signal. This means decisions about semantic content can be based on directly aggregating evidence (naive Bayes). The problem is computational: there are many possible hidden states.

The extreme application of this conditioning paradigm is classical template matching (Grenander, 1993): if the pose is rich enough to account for all non-trivial statistical variation, then even a relatively simple metric can capture the remaining uncertainty, which is basically noise. But this requires intense online computation to deform images or templates many times. One motivation of our approach is to avoid such online, global image transformations.

Similarly, the purest learning techniques, such as boosting (Viola and Jones, 2004) and convolution neural networks (LeCun et al., 2004), rely on explicitly searching through a subset of possible scales and locations in the image plane; that is, coarse scale and coarse location are not learned. Nor is invariance to illumination, usually handled at the feature level. However, invariance to other

geometric aspects of the pose, such as rotation, and to fine changes in scale and translation, are accommodated implicitly, that is, during classifier training.

On the contrary, "Part and Structure" models and other generative (model-based) approaches aim at more complex representations in terms of properties of "parts" (Li et al., 2003; Schneiderman and Kanade, 2004; Crandall and Huttenlocher, 2006). However, tractable learning and computation often require strong assumptions, such as conditional independence in appearance and location. In some cases, each part is characterized by the response of a feature detector, and the structure itself—the arrangement of parts—can either be captured by a complex statistical model, incurring severe computation in both training and testing, or by a simple model by assuming conditional independence among part locations given several landmarks, which can lead to very efficient scene parsing with the use of distance transforms. Some of these techniques do extend to highly articulated and deformable objects; see, for example, Huttenlocher and Felzenszwalb (2005). Still, modeling parts of cats (heads, ears, paws, tails, etc.) in this framework may be difficult due to the low resolution and high variation in their appearance, and in the spatial arrangements among them. Compositional models (Geman et al., 2002; Zhu and Mumford, 2006; Ommer et al., 2006) appear promising. Among these, in the "patchwork of parts" model (citepamit-trouve2007, the feature extractors are, like here, defined with respect to the pose of the object to detect, in that case a series of control points. This strategy allows for aggregating training samples with various poses through the estimation of common distributions of feature responses.

2.2 A Process of Discovery

We do not regard the hidden pose as a "nuisance" parameter, secondary to detection itself, but rather as part of what it means to "recognize" an object. In this regard, we share the view expressed in Geman et al. (2002), Crandall and Huttenlocher (2006) and elsewhere that scene interpretation should go well beyond pure classification towards rich annotations of the instantiations of the individual objects detected.

In particular, we envision detection as an organized process of discovery, as in Amit et al. (1998), and we believe that computation is a crucial issue and should be highly concentrated. Hierarchical techniques, which can accomplish focusing, are based on a recursive partitioning of the pose space (or object/pose space), which can be either ad-hoc (Geman et al., 1995; Fleuret and Geman, 2001) or learned (Stenger et al., 2006; Gangaputra and Geman, 2006). There is usually a hierarchy of classifiers, each one trained on a dedicated set of examples—those carrying a pose in the corresponding cell of the hierarchy. Often, in order to have enough data to train the classifiers, samples must be generated synthetically, which requires a sophisticated generative model.

Our work is also related to early work on hierarchical template-matching (Gavrila, 1998) and hierarchical search of pose space using branch and bound algorithms (Huttenlocher and Rucklidge, 1993), and to the cascade of classifiers in Viola and Jones (2004) and Wu et al. (2008).

Relative to the tree-based methods, we use the stationary features to aggregate data and build only one base classifier per level in the hierarchy, from which all other classifiers are defined analytically. Finally, the fully hierarchical approach avoids the dilemma of cascades, namely the sacrifice of selectivity if the pose space is coarsely explored and the sacrifice of computation if it is finely explored, that is, the cascades are dedicated to a very fine subset of poses.

3. Stationary Features

We regard the image as a random variable I assuming values in $\mathcal{I}$. The set of possible poses for an object appearing in I is $\mathcal{Y}$. We only consider geometric aspects of pose, such as the sizes of well-defined parts and the locations of distinguished points.

Let $\mathcal{Y}_1,\ldots,\mathcal{Y}_K$ be a partition of $\mathcal{Y}$. As we will see in § 4, we are interested in partitions of varying granularities for the global process of detection, ranging from rather coarse resolution (small K) to rather fine resolution (larger K), but in this section we consider one fixed partition.

For every $k = 1\ldots K$, let Y_k be a Boolean random variable indicating whether or not there is a target in I with pose in $\mathcal{Y}_k$. The binary vector $(Y_1,\ldots,Y_K)$ is denoted $\mathbf{Y}$.

In the case of merely detecting and localizing an object of fixed size in a gray-scale image of size $W \times H$, natural choices would be $\mathcal{I} = [0,1]^{WH}$ and $\mathcal{Y} = [0,W] \times [0,H]$, the image plane itself; that is, the pose reduces to one location. If the desired detection accuracy were 5 pixels, then the pose cells might be disjoint 5×5 blocks and K would be approximately $\frac{WH}{25}$. On the other hand, if the pose accommodated scale and multiple points of interest, then obviously the same accuracy in the prediction would lead to a far larger K, and any detection algorithm based on looping over pose cells would be highly costly.

We denote by $\mathcal{T}$ a training set of images labeled with the presences of targets

$$\mathcal{T} = \left\{ \left(I^{(t)}, \mathbf{Y}^{(t)} \right) \right\}_{1 \le t \le T},$$

where each $I^{(t)}$ is a full image, and $\mathbf{Y}^{(t)}$ is the Boolean vector indicating the pose cells occupied by targets in $I^{(t)}$. We write

$$\xi : \mathcal{I} \to \mathbb{R}^N,$$

for a family of N image features such as edge detectors, color histograms, Haar wavelets, etc. These are the "base features" $(\xi_1,\ldots,\xi_N)$ which will be used to generate our stationary feature vector. We will write $\xi(I)$ when we wish to emphasize the mapping and just ξ for the associated random variable. The dimension N is sufficiently large to account for all the variations of the feature parameters, such as locations of the receptive fields, orientations and scales of edges, etc.

In the next section, § 3.1, we consider the problem of "data-fragmentation", meaning that specialized predictors are trained with subsets of the positive samples. Then, in § 3.2, we formalize how fragmentation has been conventionally avoided in simple cases by normalizing the signal itself; we then propose in § 3.3 the idea of pose-indexed, stationary features, which avoids global signal normalization both offline and online and opens the way for dealing with complex pose spaces.

3.1 Data Fragmentation

Without additional knowledge about the relation between $\mathbf{Y}$ and I, the natural way to predict Y_k for each $k = 1\ldots K$ is to train a dedicated classifier

$$f_k : \mathcal{I} \to \{0,1\}$$

with the training set

$$\left\{ \left(I^{(t)}, Y_k^{(t)} \right) \right\}_{1 \le t \le T}$$

derived from $\mathcal{T}$. This corresponds to generating a single sample from each training scene, labeled according to whether or not there is a target with pose in $\mathcal{Y}_k$. This is *data-fragmentation*: training

$\mathcal{Y}$, the pose space
$\mathcal{Y}_1,\ldots,\mathcal{Y}_K$, a partition of the pose space $\mathcal{Y}$
$\mathcal{Z}$, a $W \times H$ pixel lattice
$I = \{0,\ldots,255\}^{\mathcal{Z}}$, a set of gray-scale images of size $W \times H$
I, a random variable taking values in I
Y_k, a Boolean random variable indicating if there is a target in I with pose in $\mathcal{Y}_k$
$\mathbf{Y} = (Y_1,\ldots,Y_K)$
T, the number of training images, each with or without targets
$\mathcal{T} = \left\{\left(I^{(t)},\mathbf{Y}^{(t)}\right)\right\}_{1\leq t\leq T}$, the training set
$f_k : I \to \{0,1\}$, a predictor of Y_k based on the image
Q, number of image features
$\xi : I \to \mathbb{R}^N$, a family of base image features
$\psi : \{1,\ldots,K\} \times I \to I$, an image transformation intended to normalize a given pose
$\mathbf{X} : \{1,\ldots,K\} \times I \to \mathbb{R}^Q$, a family of pose-indexed features
$\mathbf{X}(k)$, the r.v. corresponding to $\mathbf{X}(k,I)$
$g : \mathbb{R}^Q \to \{0,1\}$, a predictor trained from all the data

Table 1: Notation

f_k involves only those data which exactly satisfy the pose constraint; no synthesis or transformations are exploited to augment the number of samples available for training. Clearly, the finer the partitioning of the pose space $\mathcal{Y}$, the fewer positive data points are available for training each f_k.

Such a strategy is evidently foolhardy in the standard detection problems where the pose to be estimated is the location and scale of the target since it would mean separately training a predictor for every location and every scale, using as positive samples only full scenes showing an object at that location and scale. The relation between the signal and the pose is obvious and normalizing the positive samples to a common reference pose by translating and scaling them is the natural procedure; only one classifier is trained with all the data. However, consider a face detection task for which the faces to detect are known to be centered and of fixed scale, but are of unknown out-of-plane orientation. Unless 3D models are available, from which various views can be synthesized, the only course of action is data-fragmentation: partition the pose space into several cells corresponding to different orientation ranges and train a dedicated, range-specific classifier with the corresponding positive samples.

3.2 Transforming the Signal to Normalize the Pose

As noted above, in simple cases the image samples can be normalized in pose. More precisely, both training and scene processing involve normalizing the image through a pose-indexed transformation

$$\psi : \{1,\ldots,K\} \times I \to I.$$

The "normalization property" we desire with respect to ξ is that the conditional probability distribution of $\xi(\psi(k,I))$ given $Y_k = 1$ be the same for every $1 \leq k \leq K$.

The intuition behind this property is straightforward. Consider for instance a family of edge detectors and consider again a pose consisting of a single location z. In such a case, the transformation ψ applies a translation to the image to move the center of pose cell $\mathcal{Y}_k$ to a reference location. If

a target was present with a pose in $\mathcal{Y}_k$ in the original image, it is now at a reference location in the transformed image, and the distribution of the response of the edge detectors in that transformed image does not depend on the initial pose cell $\mathcal{Y}_k$.

We can then define a new training set

$$\left\{\left(\xi\left(\psi(k,I^{(t)})\right),Y_k^{(t)}\right)\right\}_{1\le k\le K,1\le t\le T}$$

with elements residing in $\mathbb{R}^N\times\{0,1\}$. Due to the normalization property, and under mild conditions, the new training set indeed consists of independent and identically distributed components (see the discussion in the following section). Consequently, this set allows for training a classifier

$$g:\mathbb{R}^N\to\{0,1\}$$

from which we can analytically define a predictor of Y_k for any k by

$$f_k(I)=g\left(\xi(\psi(k,I))\right).$$

This can be summarized algorithmically as follows: In order to predict if there is a target in image I with pose in $\mathcal{Y}_k$, first normalize the image with ψ so that a target with pose in $\mathcal{Y}_k$ would be moved to a reference pose cell, then extract features in that transformed image using ξ, and finally evaluate the response of the predictor g from the computed features.

3.3 Stationary Features

The pose-indexed, image-to-image mapping ψ is computationally intensive for any non-trivial transformation. Even rotation or scaling induces a computational cost of $O(WH)$ for every angle or scale to test during scene processing, although effective shortcuts are often employed. Moreover, this transformation does not exist in the general case. Consider the two instances of cats shown in Figure 3. Rotating the image does not allow for normalizing the body orientation without changing the head orientation, and designing a non-affine transformation to do so would be unlikely to produce a realistic cat image as well as be computationally intractable when done many times. Finally, due to occlusion and other factors, there is no general reason *a priori* for ψ to even exist.

Instead, we propose a different mechanism for data-aggregation based on pose-indexed features which directly assign a response to a pair consisting of an image and a pose cell and which satisfy a stationarity requirement. This avoids assuming the existence of a normalizing mapping in the image space, not to mention executing such a mapping many times online.

A **stationary feature vector** *is a pose-indexed mapping*

$$\mathbf{X}:\{1,\ldots,K\}\times I\to\mathbb{R}^Q,$$

with the property that the probability distribution

$$P(\mathbf{X}(k)=\mathbf{x}\,|\,Y_k=1),\ \mathbf{x}\in\mathbb{R}^Q \tag{1}$$

is the same for every $k=1,\ldots,K$, where $\mathbf{X}(k)$ denotes the random variable $\mathbf{X}(k,I)$.

The idea can be illustrated with two simple examples, a pictorial one in Figure 1 and a numerical one in § 3.4.

Figure 3: Aggregating data for efficient training by normalizing the pose at the image level is difficult for complex poses. For example, linear transformations cannot normalize the orientation of the body without changing that of the head.

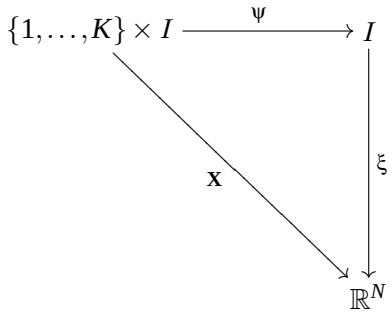
In practice, the relationship with ξ, the base feature vector, is simply that the components of the feature vector $\mathbf{X}(k)$ are chosen from among the components of ξ; the choice depends on k. In this case, we can write

$$\mathbf{X}(k) = (\xi_{\pi_1(k)}, \xi_{\pi_2(k)}, \ldots, \xi_{\pi_Q(k)}),$$

where $\{\pi_1(k), \ldots, \pi_Q(k)\} \subset \{1, \ldots, N\}$ is the ordered selection for index k. The ordering matters because we want (1) to hold and hence there is a correspondence among individual components of $\mathbf{X}(k)$ from one pose cell to another.

Note: We shall refer to (1) as the "stationarity" or "weak invariance" assumption. As seen below, this property justifies data-aggregation in the sense of yielding an aggregated training set satisfying the usual conditions. Needless to say, however, demanding that this property be satisfied exactly is not practical, even arguably impossible. In particular, with our base features, various discretizing effects come into play, including using quantized edge orientations and indexing base features with rectangular windows. Even designing the pose-indexed features to approximate stationarity by appropriately selecting and ordering the base features is non-trivial; indeed, it is the main challenge in our framework. Still, using pose-indexed features which are even approximately stationary will turn out to be very effective in our experiments with cat detection.

The contrast between signal and feature transformations can be illustrated with the following commutative diagram: Instead of first applying a normalizing mapping ψ to transform I in accordance with a pose cell k, and then evaluating the base features, we directly compute the feature responses as functions of both the image and the pose cell.



Once provided with $\mathbf{X}$, a natural training set consisting of TK samples is provided by

$$\mathcal{T}_{agg} = \left\{ \left(\mathbf{X}^{(t)}(k), Y_k^{(t)} \right) \right\}_{1 \le t \le T, 1 \le k \le K}. \tag{2}$$

Under certain conditions, the elements of this training set will satisfy the standard assumption of being independent and identically distributed. One condition, the key one, is stationarity, but technically three additional conditions would be required: 1) property (1) extend to conditioning on $Y_k = 0$; 2) the "prior" distribution $P(Y_k = 1)$ be the same for every $k = 1,\ldots,K$; 3) for each t, the samples $\mathbf{X}^{(t)}(k), k = 1,\ldots,K$, be independent. The first condition says that the background distribution of the pose-indexed features is spatially homogeneous, the second that all pose cells are *a priori* equally likely and the third, dubious but standard, says that the image data associated with different pose cells are independent despite some overlap. In practice, we view these as rough guidelines; in particular, we make no attempt to formally verify any of them.

It therefore makes sense to train a predictor $g : \mathbb{R}^Q \to \{0,1\}$ using the training set (2). We can then *define*

$$f_k(I) = g(\mathbf{X}(k,I)),\ k = 1,\ldots,K.$$

Notice that the family of classifiers $\{f_k\}$ is also "stationary" in the sense that conditional distribution of f_k given $Y_k = 1$ does not depend on k.

3.4 Toy Example

We can illustrate the idea of stationary features with a very simple roughly piecewise constant, one-dimensional signal $I(n), n = 1,...,N$. The base features are just the components of the signal itself: $\xi(I) = I$. The pose space is

$$\mathcal{Y} = \left\{ (\theta_1, \theta_2) \in \{1,\ldots,N\}^2, 1 < \theta_1 < \theta_2 < N \right\}$$

and the partition is the finest one whose cells are individual poses $\{(\theta_1,\theta_2)\}$; hence $K = |\mathcal{Y}|$. For simplicity, assume there is at most one object instance, so we can just write $Y = (\theta_1,\theta_2) \in \mathcal{Y}$ to denote an instance with pose (θ_1,θ_2). For $\mathbf{u} = (u_1,...,u_N) \in \mathbb{R}^N$, the conditional distribution of I given Y is

$$\begin{aligned} P(I = \mathbf{u} \,|\, Y = (\theta_1,\theta_2)) &= \prod_n P(I(n) = u_n \,|\, Y = (\theta_1,\theta_2)) \\ &= \prod_{n<\theta_1} \phi_0(u_n) \prod_{\theta_1 \le n \le \theta_2} \phi_1(u_n) \prod_{\theta_2 < n} \phi_0(u_n) \end{aligned}$$

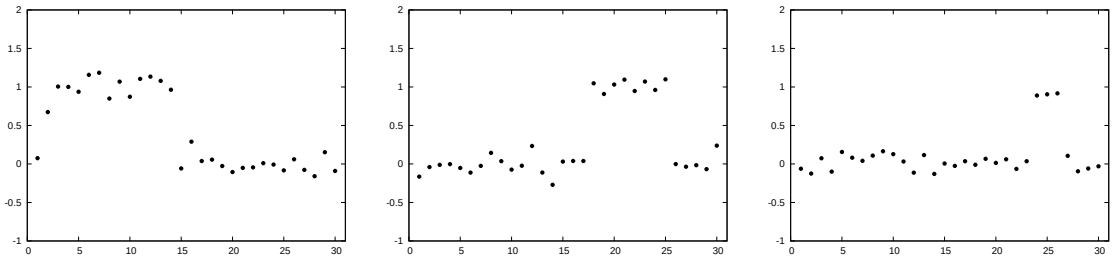


Figure 4: Examples of toy scenes

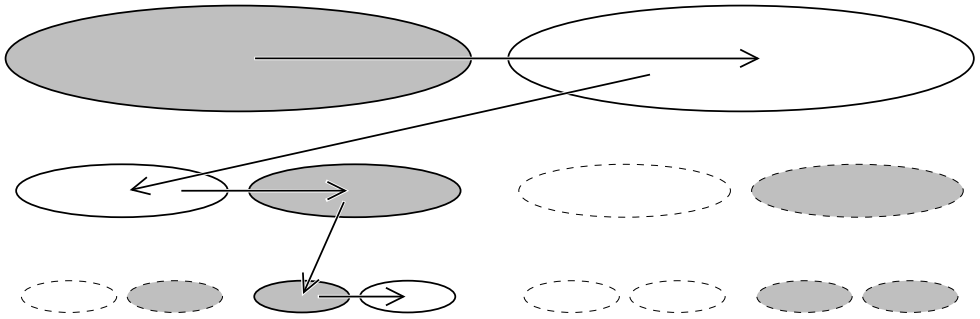


Figure 5: Hierarchical detection. Each ellipse on stands for a pose cell $\mathcal{Y}_k^{(d)}, k = 1,\ldots,K_d, d = 1,\ldots,D$. Here, $D = 3$ and $K_1 = 2$, $K_2 = 4$, $K_3 = 8$. Gray ellipses correspond to pose cells whose $f_k^{(d)}$ respond positively, and dashed ellipses correspond to pose cells whose classifiers are not evaluated during detection. As shown by the arrows, the algorithm ignores all sub-cells of a cell whose classifier responds negatively.

where ϕ_μ is a normal law with mean μ and standard deviation 0.1. Hence the signal fluctuates around 0 on the "background" and around 1 on the target, see Figure 4.

We define a four-dimensional pose-indexed feature vector taking the values of the signal at the extremities of the target, that is

$$\mathbf{X}((\theta_1,\theta_2),I) = (I(\theta_1-1),I(\theta_1),I(\theta_2),I(\theta_2+1)).$$

Clearly,

$$P(\mathbf{X}(\theta_1,\theta_2) = (x_1,x_2,x_3,x_4)\,|\,Y_{\theta_1,\theta_2} = 1) = \phi_0(x_1)\phi_1(x_2)\phi_1(x_3)\phi_0(x_4)$$

which is not a function of θ_1,θ_2. Consequently, $\mathbf{X}$ is stationary and the common law in (1) is $\phi_0 \times \phi_1 \times \phi_1 \times \phi_0$.

4. Folded Hierarchies

We have proposed normalizing the samples through a family of pose-indexed features instead of whole image transforms in order to avoid fragmentation of the data. Since only one classifier must be built for any partition of the pose space, and no longer for every cell of such a partition, neither

the cost of learning nor the required size of the training set grows linearly with the number K of pose cells in the partition. However, one main drawback remains: We must still visit all the pose cells online, which makes the cost of scene processing itself linear in K.

A natural strategy to address computational cost is an hierarchical search strategy based upon a recursive partitioning of $\mathcal{Y}$. As in previous work (Fleuret and Geman, 2001; Gangaputra and Geman, 2006), there is a succession of nested partitions of increasing resolution and a binary classifier assigned to each cell. Given such a hierarchy, the detection process is adaptive: a classifier is evaluated for a certain pose cell only if all the classifiers for its ancestor cells have been evaluated and responded positively.

Note: This is *not* a decision tree, both in terms of representation and processing. The hierarchy recursively partitions the space of hidden variables not the feature space, and the edges from a node to its children do not represent the possible values of a node classifier. Moreover, during processing, a data point may traverse many branches at once and may reach no leaves or reach many leaves.

The crucial difference with previous work is that, using stationary features, only one classifier must be trained for each level, not one classifier for each cell. In essence, the hierarchy is "folded" (like a fan) for training: The entire learning strategy described in §3 is repeated for each level in the hierarchy. This is quite straightforward and only summarized below.

Consider a sequence of partitions of $\mathcal{Y}$

$$\left\{\mathcal{Y}_1^{(d)},\ldots,\mathcal{Y}_{K_d}^{(d)}\right\},\quad 1\leq d\leq D,$$

for which any cell $\mathcal{Y}_k^d$ for $k=1,\ldots,K_{d+1}$, is a (disjoint) union of cells at the next level $d+1$. Consequently, we can identify every $\mathcal{Y}_k^{(d)}$ with the node of a multi-rooted tree: A leaf node for $d=D$ and an internal node otherwise. A three-level hierarchy is shown in Figure 5.

Given such a pose hierarchy, we can construct a scene parsing algorithm aimed at detecting all instances of objects at a pose resolution corresponding to the finest partition. Again, the processing strategy is now well-known. This algorithm has the desirable property of concentrating computation on the ambiguous pose-image pairs.

Let $Y_k^{(d)}$ denote a Boolean random variable indicating whether or not there is a target in I with pose in $\mathcal{Y}_k^{(d)}$ and let $\mathbf{X}^{(d)}$ denote a pose-indexed feature vector adapted to the partition $\{\mathcal{Y}_1^{(d)},\ldots,\mathcal{Y}_{K_d}^{(d)}\}$. For each level d, we train a classifier $g^{(d)}$ exactly as described in §3.3, and define a predictor of $Y_k^{(d)}$ by

$$f_k^{(d)}(I)=g^{(d)}\left(\mathbf{X}^{(d)}(k,I)\right).$$

The hierarchy is "unfolded" for testing and the predictors are evaluated in an adaptive way by visiting the nodes (cells) according to breadth-first "coarse-to-fine" search. A classifier is evaluated if and only if all its ancestors along the branch up to its root have been evaluated and returned a positive response. In particular, once a classifier at a node responds negatively, none of the descendant classifiers are ever evaluated. The result of the detection process is the list of leaves which are reached and respond positively. In this way, pose cells corresponding to obvious non-target regions such as flat areas are discarded early in the search and the computation is invested the ambiguous areas, for example, parts of images with "cat-like" shape or texture.

5. Base Classifier

As described in §4, given a family of pose-indexed features and a hierarchical partitioning of the pose space, we build a binary classifier $g^{(d)}$ for each level d in the hierarchy, trained from a set of examples of the type described in §3.3. In this section we describe that classifier, dropping the superscript d for clarity. The actual parameter values we used for the experiments on cat detection are given in §6.5.

Evidently, inducing such a mapping g is a standard machine learning problem. A simple candidate is a thresholded linear combination of V stumps trained with Adaboost (Freund and Schapire, 1999):

$$g(x) = \begin{cases} 1 & \text{if } \sum_{i=1}^{V} \alpha_i 1_{\{x^{\delta_i} \geq \tau_i\}} \geq \rho \\ 0 & \text{otherwise.} \end{cases}$$

Here, x^j is the j'th coordinate of the feature vector.

For any given true positive rate η, the threshold ρ in g, and more generally the thresholds $\rho^{(d)}$ in $g^{(d)}, 1 \leq d \leq D$, are chosen to achieve on a validation set a targeted decreasing sequence of true positive rates yielding η.

To select the stumps, that is the α_i, δ_i and τ_i, special attention must be given to the highly unbalanced populations we are dealing with. Of course in our detection problem, the prior distribution is very skewed, with an extremely low probability of the presence of a target at a pose picked at random. Correspondingly, the number of samples we have from the positive population (cats in our case) is orders of magnitude smaller than the number of samples we can easily assemble from the negative population. Still, any tractable sampling of the negative population is still too sparse to account for the negative sub-population which lives close to positive examples. To address these issues, we propose a variation the standard weighting-by-sampling in order to approximate standard Adaboost using a training set containing one million negative examples.

The popular cascade approach handles that dilemma with bootstrapping: training each level with a sample of negative examples which survive the filtering of the previous classifiers in the cascade. In this way the sampling is eventually concentrated on the "difficult" negative samples. This is similar in practice to what boosting itself is intended to do, namely ignore easily classified samples and concentrate on the difficult ones. We avoid the complexity of tuning such a cascade by using all the negative examples through an asymmetric, sampling-based version of standard boosting. This provides an excellent approximation to the exact weighting for a fraction of the computational cost.

When picking a stump we approximate the weighted error with an error computed over all positive samples and a *random subset of negative samples* drawn according to the current boosting weights. Hence, we keep the response of the strong classifier up-to-date on $S \simeq 10^6$ samples, but we pick the optimal weak learners at every step based on $M \simeq 10^4$ samples.

More precisely, at a certain iteration of the boosting procedure, let ω_s denote the weight of sample $s = 1, \ldots, S$, let $Y_s \in \{0,1\}$ be its true class, and let

$$\omega_{neg} = \sum_s \omega_s \mathbf{1}_{\{Y_s = 0\}}$$

be the total weight of the negative samples. We sample independently M indices $S_1,\ldots,S_M$ in $\{1,\ldots,S\}$ according to the negative sample density

$$P(S_m = s) = \frac{\omega_s \mathbf{1}_{\{Y_s=0\}}}{\omega_{neg}},\ m = 1,\ldots,M.$$

Then we re-weight the training samples as follows:

$$\omega'_s = \begin{cases} \omega_s & \text{if } Y_s = 1 \\ \omega_{neg}\dfrac{\|\{m : S_m = s\}\|}{M} & \text{otherwise.} \end{cases}$$

This can be seen as an approximation to the distribution on the full training set obtained by (1) keeping all positive samples with their original weights, and (2) selecting a random subset of negative samples according to their original weights, and giving them a uniform weight.

However, sampling the negative examples at every boosting step is computationally very expensive, mainly because it requires loading into memory all the training images, and extracting the edge features at multiple scales. Consequently, we decompose the total number of stumps V into B blocks of U stumps, and run this sampling strategy at the beginning of every block. We also sample a subset of R features among the millions of features available at the beginning of every block. Hence, our overall learning process can be seen as a standard boosting procedure decomposed into B blocks of U steps. At the beginning of every block, we sample M negative samples among S according to their current boosting weight and we sample R features uniformly among the Q to consider. We then run U boosting steps based on these features and these training samples.

This process ensures that any sample for which the classifier response is strongly incorrect will eventually be picked. In our experiments we sample ten negative examples for every positive example. From a computational perspective this sampling is negligible as it only accounts for about 1% of the total training time.

6. Cat Detection

We now specialize everything above to cat detection. The original training images and available ground truth are described in §6.1. Then, in §6.2, we define a family of highly robust, base image features based on counting edge frequencies over rectangular areas, and in §6.3 we propose a way to index such features with the pose of a cat defined by its head location and scale and belly location. The specific pose cell hierarchy is described in §6.4 and choice of parameters for the classifiers in §6.5. Finally, the results of our experiments are presented in §6.6, including the similarity criteria used for performance evaluation, the post-processing applied to the raw detections in order to reduce "duplicate" detections and the manner of computing ROC curves.

6.1 Cat Images and Poses

The cat images were randomly sampled from the web site RateMyKitten[1] and can be downloaded from `http://www.idiap.ch/folded-ctf`. Images of cluttered scenes without cats, mostly home interiors, were sampled from various web sites. The complete database we are using has 2,330 images containing a total of 1,988 cats.

1. Web site can be found at `http://www.ratemykitten.com`.

Figure 6: Each target is labeled by hand with a pose $(h,r,b) \in \mathcal{Z} \times \mathbb{R}^+ \times \mathcal{Z}$ specifying the head location, the head radius and the belly location.

Each cat was manually annotated with one circle roughly outlining the head, from which the head size (diameter) and head location (center) are derived, and one point placed more or less, quite subjectively, at the center of mass, which we have referred to as the "belly" location (see Figure 6). Hence, the pose of a cat is (h,r,b) where h is the location in the image plane $\mathcal{Z}$ of the center of the head, r its radius, and b is the belly location.

For each experiment, we split this database at random into a training set containing 75% of the images and a test set containing the other 25%. Two-thirds of the training set are used for choosing the weak learners and one-third for the thresholds.

6.2 Base Image Features

As described in §3, the pose-indexed features are defined in terms of base image features. First, an image is processed by computing, at every location $z \in \mathcal{Z}$, the responses of eight edge-detectors similar to those proposed in Amit et al. (1998) (see Figures 7 and 8), but at three different scales, ending up with 24 Boolean features $e_1(z), \ldots, e_{24}(z)$ corresponding to four orientations and two polarities. In addition, we add a variance-based binary test $e_0(z)$ which responds positively if the variance of the gray levels in a 16×16 neighborhood of z exceeds a fixed threshold. Our features are based on counting the number of responses of these 25 detectors over a rectangular areas (Fan, 2006), which can be done in constant time by using 25 integral images (Simard et al., 1999).

From these edge maps and the raw gray levels we define the following three types of base image features:

1. **Edge proportion:** The proportion of an edge type in a rectangular window. Given a rectangular window W and an edge type $\lambda \in \{0,1,\ldots,24\}$, the response is the number of pixels z in W for which $e_\lambda(z) = 1$, divided by the total number of pixels in W if $\lambda = 0$ or by the number of pixels in W for which $e_0(z) = 1$ if $\lambda > 0$.

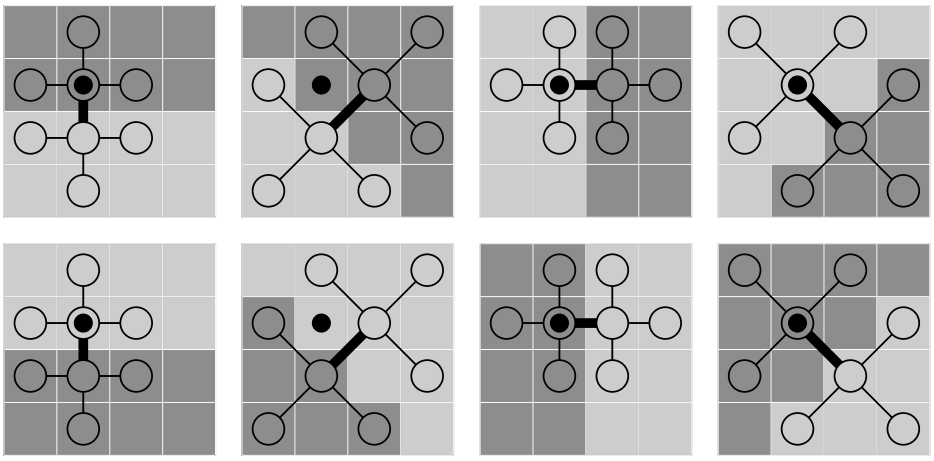


Figure 7: Our edge-detectors: For each of four orientations and two polarities, an edge is detected at a certain location (the dark circle) if the absolute difference between the intensities of the two pixels linked by the thick segment is greater than each of the six intensity differences for pixels connected by a thin segment.

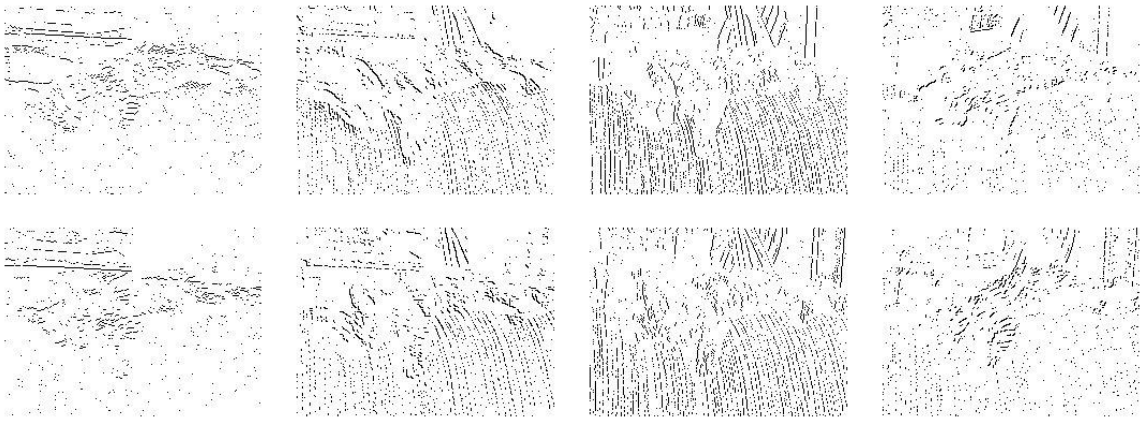


Figure 8: Result of the edge detector. Each one of the eight binary images in the two bottom rows corresponds to one orientation of the edge detectors of Figure 7.

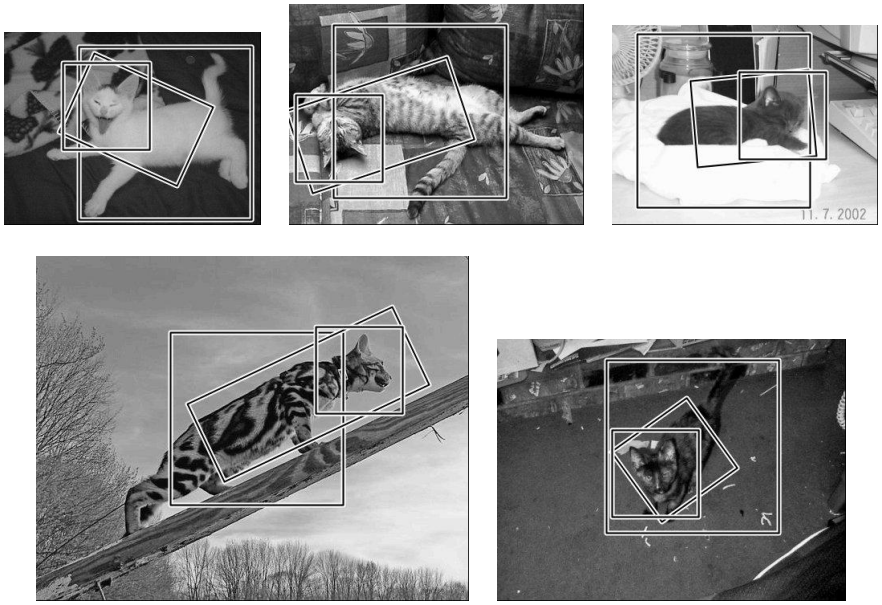


Figure 9: From the centroid of any pose cell, we define three reference frames: The *head frame* is centered on the head center, of size twice the head size; the *belly frame* is centered on the belly and of size four times the head size; the *head-belly frame* is centered on the middle point between the head and the belly, of height twice the head size, of width twice the distance between the head and the belly, and is tilted accordingly.

2. **Edge orientation histogram distance:** Given again two rectangular windows W_1 and W_2, and a scale s, the response is the L^1 norm between the empirical eight-bin histograms of orientations corresponding to the eight edge detectors at scale s.

3. **Gray-scale histogram distance:** Given two rectangular windows W_1 and W_2, the response is the L^1 norm between the sixteen-bin empirical histograms of gray-scales for the two windows.

The rational behind the features of type 1 is to endow the classifiers with the ability to check for the presence of certain pieces of outlines or textures. The motivation for types 2 and 3 is to offer the capability of checking for similarity in either edge or gray-scale statistics between different parts of the image, typically to check for a silhouette in the case of very blurry contours. Some examples of features actually picked during the training are shown in Figures 10 and 11.

6.3 Indexing Features by Pose

As formalized in §3.3, a pose-indexed feature is a real-valued function of both a pose cell and an image. The features described in the previous section are standard functionals of the image alone. Since the response of any of them depends on counting certain edge types over rectangular windows in the image, we construct our family of pose-indexed features indirectly by indexing both the edge types and the window locations with the pose cell.

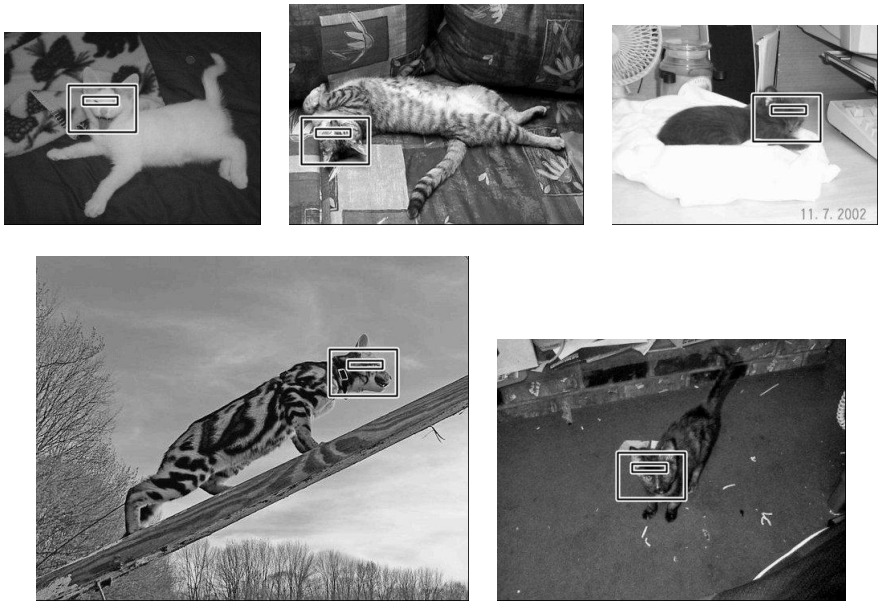


Figure 10: Registration on the true poses of the first feature selected in the HB detector (see §6.5), which compares edge orientation histograms. Both windows are defined relative to the head frame.

For any pose cell index k, we compute the average head location $h = h_k$, the average belly location $b = b_k$, and the average head radius $r = r_k$ of the pose cell $\mathcal{Y}_k$. From these parameters we compute three reference frames, as shown on Figure 9:

1. The **head frame** is a square centered on h and of size $4r$.
2. The **belly frame** is a square centered on b and of size $8r$.
3. The **head-belly frame** is a rectangle centered on the midpoint of h and b, tilted accordingly, and of height $4r$ and width twice $\|h-b\|$.

Note that the definition of such a frame actually involves the definition of a vector basis, hence an orientation. The three types of frames are oriented according to the relative horizontal locations of the head and belly of the cat, so a reflection around a horizontal axis of the image, hence of the cat pose, would move the points defined in these frames consistently.

We add to the parameterization of each feature window a discrete parameter specifying in which of these three reference frames the window is defined. Windows relative to the head or the belly frame are simply translated and scaled accordingly. Windows relative to the head-belly frame are translated so that their centers remain fixed in the frame, and are scaled according to the average of the height and width of the frame. See Figures 10 and 11.

Finally, we add another binary flag to windows defined in the head-belly frame to specify if the edge detectors are also registered. In that case, the orientation of the edges is rotated according to the tilt of the head-belly frame.

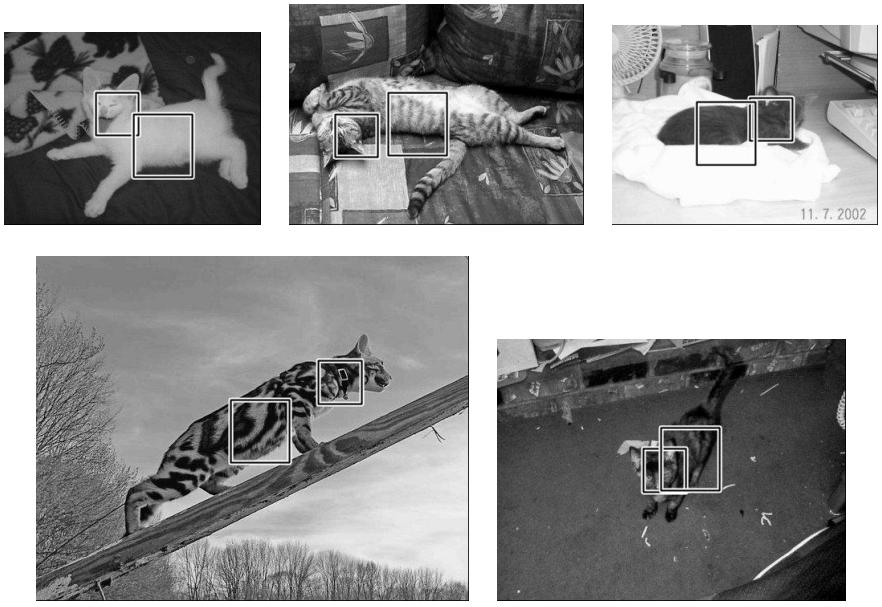


Figure 11: Registration on the true poses of the third feature selected in the second level of the HB detector (see §6.5), which compares grayscale histograms. One window is relative to the head-belly frame, and the second one to the belly frame.

6.4 Hierarchy of Poses

We only consider triples (h,r,b) which are consistent with the relative locations seen on the training set. For instance, this discards poses with very large ratios $\|h-b\|/r$. However, h and b may be very close together, for example when the belly is behind the head, or very far apart, for example when the cat is stretched out. Hence the full pose space is $\mathcal{Y} \subset \mathcal{Z} \times \mathbb{R}^+ \times \mathcal{Z}$.

We use a hierarchy with only $D = 2$ levels in order to concentrate on folded learning with stationary features. The first level $\{\mathcal{Y}_1^{(1)}, \ldots, \mathcal{Y}_{K_1}^{(1)}\}$ is based on first restricting the head radius to $[25, 200]$, and on splitting that domain into 15 sub-intervals of the form $[r, 2^{1/5} r]$. For each such scale interval, we divide the full lattice $\mathcal{Z}$ into non-overlapping regular squares of the form $[x_h, x_h + r/5] \times [y_h, y_h + r/5]$. This procedure creates $K_1 \simeq 50,000$ head parameter cells $[x_h, x_h + r/5] \times [y_h, y_h + r/5] \times [r, 2^{1/5} r]$ for a 640×480 image. For any such cell, the admissible domain for the belly locations is the convex envelope of the belly locations seen in the training examples, normalized in location and scale with respect to the head location and radius.

The second level $\{\mathcal{Y}_1^{(2)}, \ldots, \mathcal{Y}_{K_2}^{(2)}\}$ is obtained by splitting the belly location domain into regular squares $[x_b, x_b + r/2] \times [y_b, y_b + r/2]$. There are $\simeq 500$ such belly squares, hence the total number of pose cells in the second level is $K_2 \simeq 2.5 \times 10^7$.

The top-left illustration in Figure 12 depicts the cells in the first level of the hierarchy as open circles and cells in the second level as black dot connected to an open circle "kept alive" during processing the first level. More specifically, as shown in Figure 12, the algorithmic process corresponding to this two-level hierarchy is as follows:

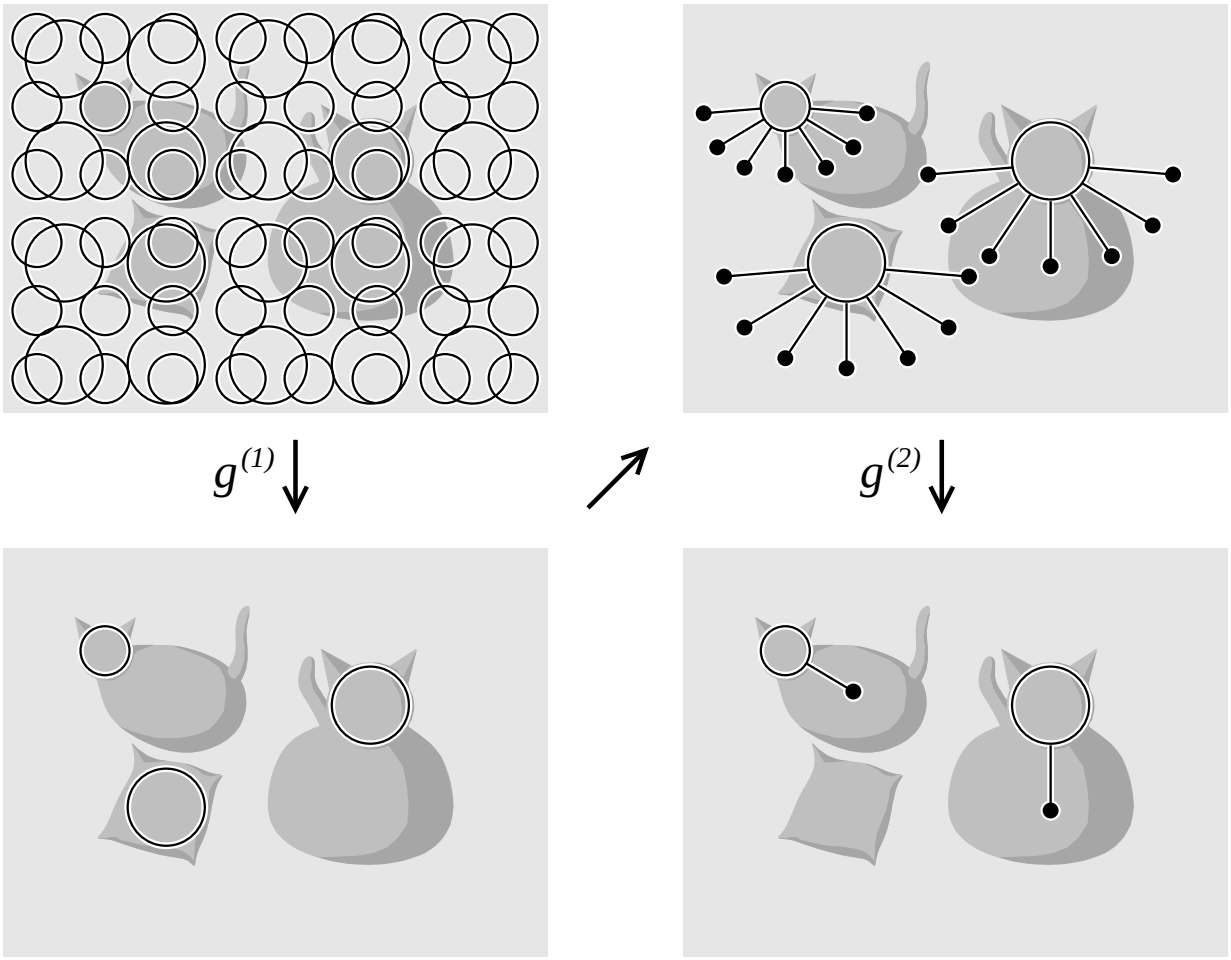


Figure 12: Parsing a scene with a two-level hierarchy to find cats: First, a classifier $g^{(1)}$ is evaluated over a sublattice of possible head locations and all alarms above a very low threshold are retained. Then a classifier $g^{(2)}$ is evaluated for each pair of head-belly locations on a sublattice consistent with the retained head alarms and with observed statistics about joint head-belly locations. For clarity, the depicted discretization of the pose space is idealized, and far coarser than in the actual experiments; for an image of size 640×480 pixels, we consider $\simeq 50,000$ head pose cells and $\simeq 2.5 \times 10^7$ head-belly pose cells.

1. The first stage loops over a sublattice of possible head locations and scales in the scene, evaluates the response of the appropriate first-level classifier and retains all alarms using a very low (i.e., conservative) threshold.

2. The second stage visits each location and scale tagged by the first stage, scans a sublattice of all "consistent" belly locations (all those actually observed on training images) and evaluates an appropriate second-level classifier for every such candidate pair of locations.

6.5 Detectors

Whereas our aim is to detect both the head and the body, detecting the head alone is similar to the well-studied problem of detecting frontal views of human faces. As stated earlier, if the pose reduces to a single position, data-aggregation is straightforward by translating either whole images or features. Still, detecting cat heads is a logical first step in trying to find cats since the head is clearly the most stable landmark and the part of the cat with the least variation, assuming of course that the head is visible, which is the case with our data (for the same reason that family photographs display the faces of people). Moreover, comparing the performance of varying strategies (field of view, "checking" for the belly separately, demanding "consistency", etc.) provides some insight on the nature of the problem and serves as a simple way of demonstrating the power of the base feature set and the asymmetrical weighting by sampling. Detecting heads alone does not, however, expose the full strength of the folded hierarchy; for that we need to address the harder task of accurately estimating (h,r,b) for the visible cats, our core objective, and for which we will compare our pose-indexed method with a more standard parts-based detector.

In all the experiments we present, the classifiers are trained as described in §5, with $B = 25$ blocks of $U = 100$ stumps (thresholded features), and we optimize over a sample of $R = 10,000$ pose-indexed features in every such block. The total number of negative samples we consider is $S \simeq 10^6$, and we sample $M \simeq 10^4$ of these per block.

In measuring performance, we consider the two following detections strategies:

- **H+B** is a standard parts detector, implemented adaptively. The "+" between H and B indicates that the two part detectors are trained separately.

  The first level classifier $g^{(1)}$ can only use pose-indexed feature defined relatively to the head frame and the second level classifier $g^{(2)}$ can only use pose-indexed features defined relatively to the belly frame. Since that second-level detector is designed not to exploit the information in the joint locations of the head and belly, the frames here have fixed orientation, and reflecting the cat pose horizontally would move but not invert the frames. See §6.3 for details about the orientations of the frames.

- **HB** is the hierarchical detector based on the two-level hierarchy and folded learning.

  The difference with H+B is that HB uses stationary features in the second level which can be defined relatively to any of the three reference frames (head, belly or head-belly) in order to take into account the position of the head in searching for the belly. For instance, a pose-indexed features in this detector could compare the texture between a patch located on the head and a patch located on the belly.

6.6 Results

In order to be precise about what a constitutes a true detection, we define two criteria of similarity. We say that two poses (h,r,b) and (h',r',b') **collide** if (1) The head radii are very similar: $1/1.25 \leq r/r' \leq 1.25$; and (2) Either the head or belly locations are close: $\min(\|h-h'\|, \|b-b'\|) \leq 0.25\sqrt{rr'}$. And we will say that two poses are **similar** if (1) The head radii are similar: $1/1.5 \leq r/r' \leq 1.5$; (2) the head locations are nearby each other: $\|h-h'\| \leq 2\sqrt{rr'}$; and (3) the belly locations are nearby each other: $\|b-b'\| \leq 4\sqrt{rr'}$. See Figure 13.

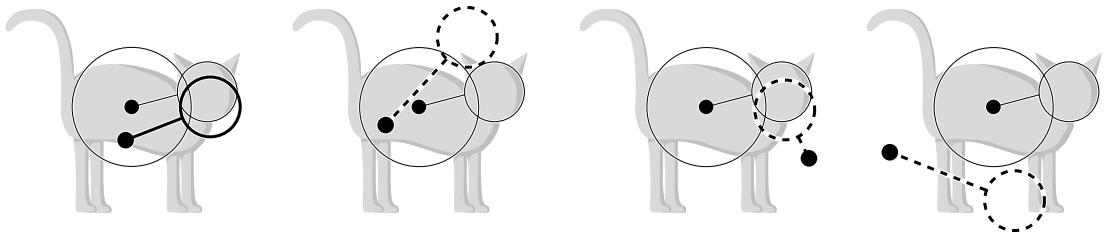


Figure 13: Two alarms are considered as similar if the head radii are similar and if, as shown on this figure, the distance between the two head locations is less than the average head radius, and if the distance between the belly locations is less than twice the average head radius. See §6.6. Based on that criterion, if the true pose is the one shown in thin lines and the thick poses are detections, only the leftmost one would be counted as a true hit. The three others, shown in dashed lines, would be counted as false alarms.

Given these two criteria, the alarms kept after thresholding the classifier responses are post-processed with a crude clustering. We visit the alarms in the order of the response of the detector, and for each alarm we remove all others that collide with it. Then we visit these surviving alarms again in the order of the response and for each alarm we remove all the other alarms which are similar.

The procedure we use to produce ROC curves is the following. We run ten rounds in which the training and test images are selected at random, and in each round we estimate the classifier thresholds for achieving ten different true-positive rates η (see § 5). Hence, we generate 100 pairs of rates, each consisting of a true-positive rate and an average number of false alarms per image. An alarm is counted as true positive if there exists a cat in the image with a similar pose according to the criterion described above.

The error rates in Figure 2 and Table 2 demonstrate the power of conditioning on the full pose. Using stationary features to build classifiers dedicated to fine cells allows the search for one part to be informed by the location of the other, and allows for consistency checks. This is more discriminating than checking for individual parts separately. Indeed, the error rates are cut be a factor of roughly two at very high true-positive rates and a factor of three at lower true-positive rates. It should be emphasized as well that even the weaker ROC curve is impressive in absolute terms, which affirms the efficacy of even the naive stationary features used by the H+B detector and the modified boosting strategy for learning.

An example of how features selected in the second-level of the HB classifier exploit the full pose can be seen in Figure 11. Such a feature allows the HB detector to check for highly discriminating properties of the data, such as the continuity of appearance between the head and the belly, or discontinuities in the direction orthogonal to the head-belly axis.

More then two-thirds of the false positives are located on or very near cats; see Figure 14. Such false positives are exceedingly difficult to filter out. For instance, a false head detection lying around or on the belly will be supported by the second-level classifier because the location of the true belly will usually be visited.

TP	H+B	HB
90%	12.84	5.85
80%	3.53	1.63
70%	1.35	0.50
60%	0.61	0.23
50%	0.33	0.12
40%	0.18	0.06
30%	0.10	0.03

Table 2: Average number of false alarms per images of size 640×480 vs. the true positive rate for the head-belly detection, as defined by the similarity criterion of §6.6 and Figure 13.

Finally, we performed a similar experiment by testing the classifiers trained on the Ratemykitten data set on a sample of cat images chosen from the PASCAL VOC2007 challenge set images.[2] The PASCAL data set was assembled for evaluating methods for *classification*, that is, labeling an entire image according to one of the object categories, rather than methods for object detection and localization. There are 332 cat images in the PASCAL set; our test set consists of those 201 images for which the body is at least partially visible. This provides an even more challenging test set than the images from Ratemykitten and the performance of our classifier is somewhat reduced. For instance, at a true positive rate of 51%, the average number of false alarms per image of size 640×480 is 0.9. The results on a random sample of twenty of the 201 test images is shown in Figure 15.

7. Conclusion

We have presented a novel detection algorithm for objects with a complex pose. Our main contribution is the idea of stationary, pose-indexed features, a variation on deformable templates without whole image transforms. This makes it possible to train pose-specific classifiers without clustering the data, and hence without reducing the number of training examples. Moreover, combining simultaneous training with a sequential exploration of the pose space overcomes the main drawback of previous coarse-to-fine strategies, especially for going beyond scale and translation. Unlike in earlier variations, graded, tree-structured representations can now be learned efficiently because there is only one classifier to train per level of the hierarchy rather than one per node.

We have illustrated these stationary features by detecting cats in cluttered still images. As indicated earlier, the data are available at `http://www.idiap.ch/folded-ctf`. We chose boosting with edge and intensity counts, but any base learning algorithm and any flexible base feature set could be used. Indeed, the framework can accommodate very general features, for instance the average color or average response of any local feature in an area defined by the pose. The resulting algorithm is a two-stage process, first visiting potential head locations alone and then examining additional aspects of the pose consistent with and informed by candidate head locations.

In principle, our approach can deal with very complex detection problems in which multiple objects of interest are parametrized by a rich hidden pose. However, two basic limitations must

2. Website can be found at `http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/`.

Figure 14: Detection results with stationary features and a folded two-level hierarchy on scenes picked uniformly at random in the RateMykitten test set, with a true-positive rate of 71%. The circle shows the estimated head size and location, and the dot the estimated belly location.

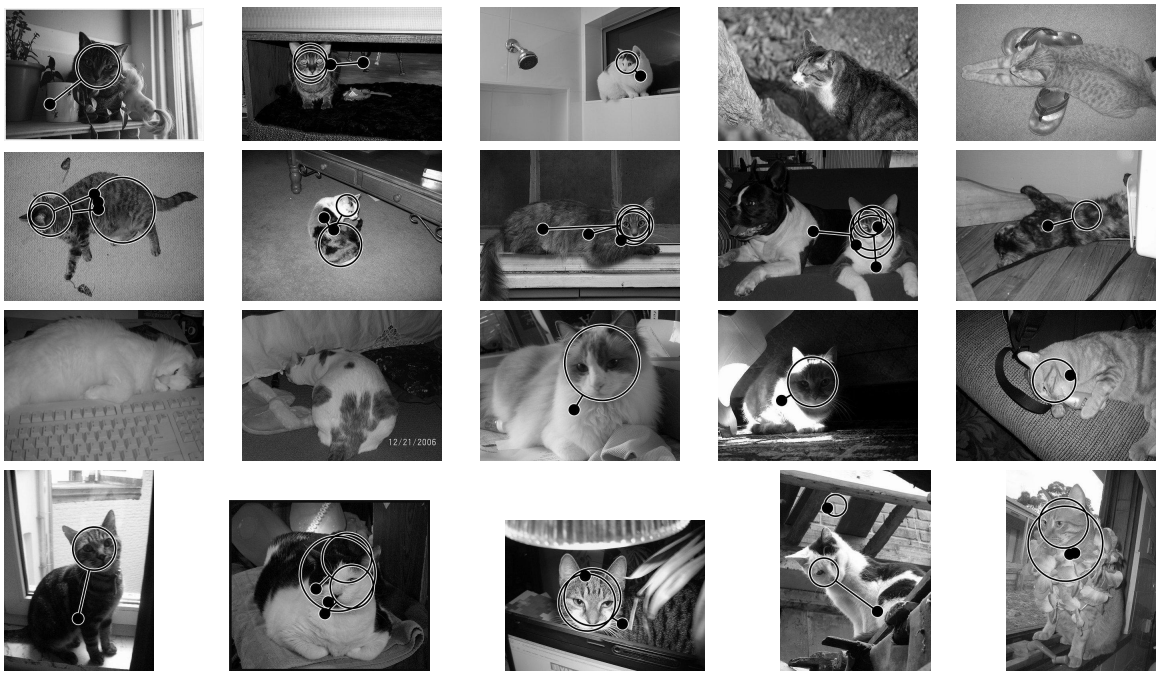


Figure 15: Detection results with stationary features and a folded two-level hierarchy on scenes picked uniformly at random in the PASCAL VOC2007 challenge test set, with a true-positive rate of 50%. The circle shows the estimated head size and location, and the dot the estimated belly location.

first be addressed. The first is the design of adequate stationary features. Whereas difficult, this is far simpler than the search for full geometric invariants. Since the hidden state is explicitly examined in traversing the hierarchy, there is no need to integrate over all possible values of the hidden quantities. The second difficulty is labeling a training set with rich ground truth. One way to tackle this problem is by exploiting other information available during training, for instance temporal consistency if there are motion data. Our viewpoint is that small, richly annotated, training sets are at least as appealing for general learning as large ones with minimal annotation.

Acknowledgments

The work of FF was partially supported by the Swiss National Science Foundation under the National Centre of Competence in Research (NCCR) on "Interactive Multimodal Information Management" (IM2). The work of DG was partially supported by the National Science Foundation under grants NSF 0427223 and NSF 0625687, and by the Office of Naval Research under contract N00014-07-1-1002.

We are also extremely grateful to the reviewers for their thoughtful suggestions, as well as to the web site `http://www.ratemykitten.com`, and to Harrison Page in particular, for providing us with the remarkable set of cat images.

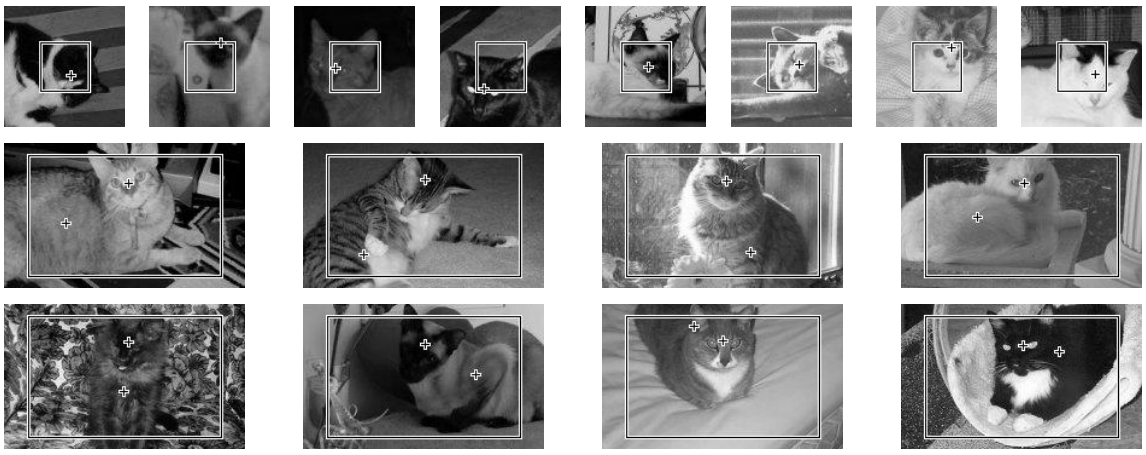


Figure 16: A few positive examples picked uniformly at random in the simplified setting of the motivational experiment. Top row: samples from the head experiments. Bottom two rows: samples from the head-belly experiments. The crosses depict the head and belly centers provided on the training data. The boxes show the admissible pose domain $\mathcal{Y}$.

Appendix A. Quantifying Trade-offs

We summarize two series of experiments designed to study the impact on accuracy of data-fragmentation, with and without controlling for total online computation. In both series the goal is to predict the presence of a target with high accuracy in pose. A more detailed account of these experiments can be found in Fleuret and Geman (2007).

A.1 Settings

Since training with fragmentation is not feasible for any complete partition of a complex pose space at a realistic resolution, the images we consider in these experiments have been cropped from the original data set so that the pose space $\mathcal{Y}$ is already strongly constrained.

In the first series of experiments the target pose is the center of the cat head, constrained to $\mathcal{Y} = [-20,20] \times [-20,20]$ in a 100×100 image. It is this pose space that will be investigated at different resolutions. The top row of Figure 16 shows a few of these scenes with a target. In the second series of experiments the pose is the *pair of locations* (h,b) for the head and belly, constrained to $\mathcal{Y} = ([0,5] \times [0,5]) \times ([-80,80] \times [-20,80])$ in a 200×140 image centered at the square. The two bottom rows of Figure 16 show a few of these scenes with a target.

In both series, our objective is to compare the performance of classifiers when the training data are either fragmented or aggregated and when the computational cost is either equalized or not. More precisely, we consider three partitions of $\mathcal{Y}$ into $K = 1$, 4 and 16 pose cells. In each series, we build four detection systems. Three of them are trained under data-fragmentation at the three considered resolutions, namely $K = 1$, $K = 4$ or $K = 16$ pose cells. The fourth classifier is trained with the pose-indexed, stationary features at the finest resolution $K = 16$. The stationary features are based on the head frame alone for the head experiments, and on both the head frame and the head-belly frame for the head-belly experiments.

The computational cost for evaluating one such classifier is proportional to the number of stumps it combines. In the particular case of boosting, a classifier combining only a fixed number of weak learners is still effective, and hence, unlike many discriminative methods, computation is easy to control. This motivates a very simple strategy to equalize the cost among experiments: We simply control the total number of feature evaluations.

As a measure of performance, we estimate the number of false alarms for any given true positive rate. In order to compare results across resolutions, the labeling of detections as true or false positives occurs at the coarsest resolution. For simplicity, for the head-belly case, we only score the estimated head location.

A.2 Results

The results demonstrate the gain in performance in constraining the population provided there is no fragmentation of the data. In the head experiments, even with fragmentation, higher resolution results in fewer false alarms. The improvement is marginal at high true positive rates, but increases to two-fold for a true positive rate of 70%. This is not true for the head-belly experiments, where sixteen pose cells do worse than four, with or without cost equalization, which can be explained to some extent by the lower variation in the appearance of cat heads than full cat bodies, and hence fewer samples may be sufficient for accurate head detection.

As expected, without controlling the on-line computational cost, aggregation with stationary features is more discriminating than the fragmented classifiers in both experiments and at any true positive rate, reducing the false positive rate by a factor of three to five. Still, the performance of the classifiers when cost is equalized shows the influence of computation in this framework: at the finest resolution, the number of false alarms in the head experiments increases by a factor greater than four at any true-positive rate, and by two orders of magnitude in the head-belly experiments.

These results also demonstrate the pivotal role of computation if we are to extend this approach to a realistically fine partition of a complex pose space. Consider an image of resolution 640×480 and a single scale range for the head. Obtaining an accuracy in the locations of the head and the belly of five pixels requires more than 7×10^6 pose cells. Investing computation *uniformly* among cells is therefore hopeless, and argues for an adaptive strategy able to distribute computation in a highly special and uneven manner.

The conclusions drawn can be summarized in two key points:

1. **The need for data-aggregation**: *Dealing with a rich pose by training specialized predictors from constrained sub-populations is not feasible, both in terms of offline computation and sample size requirements. Aggregation of data using stationary features appears to be a sound strategy to overcome the sample size dilemma as it transfers the burden of learning to the design of the features.*

2. **The need for adaptive search**: *If fragmentation can be avoided and a single classifier built from all the data and analytically transformed into dedicated classifiers, the computation necessary to cover a partition of a pose space of reasonable accuracy is not realistic if the effort is uniformly distributed over cells.*

As indicated, stationary features provide a coherent strategy for dealing with data-aggregation but do not resolve the computational dilemma resulting from investigating many possible poses during scene processing. Hierarchical representations largely do.

References

Y. Amit, D. Geman, and B. Jedynak. Efficient focusing and face detection. In *Face Recognition: From Theory to Applications*. Springer Verlag, 1998.

D. J. Crandall and D. P. Huttenlocher. Weakly supervised learning of part-based spatial models for visual object recognition. In *European Conference on Computer Vision*, pages 16–29, 2006.

X. Fan. *Learning a Hierarchy of Classifiers for Multi-class Shape Detection*. PhD thesis, Johns Hopkins University, 2006.

F. Fleuret and D. Geman. Coarse-to-fine face detection. *International Journal of Computer Vision (IJCV)*, 41(1/2):85–107, 2001.

F. Fleuret and D. Geman. Stationary features and cat detection. Technical Report 07-56, IDIAP Research Institute, October 2007.

Y. Freund and R. E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, 1999.

S. Gangaputra and D. Geman. A design principle for coarse-to-fine classification. In *Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1877–1884, 2006.

D.M. Gavrila. Multi-frame hierarchical template matching using distance transforms. In *International Conference on Pattern Recognition*, 1998.

S. Geman, K. Manbeck, and E. McClure. Coarse-to-fine search and rank-sum statistics in object recognition. Technical report, Brown University, 1995.

S. Geman, D. F. Potter, and Z. Chi. Composition systems. *Quarterly of Applied Mathematics*, LX: 707–736, 2002.

U. Grenander. *General Pattern Theory*. Oxford U. Press, 1993.

D. Huttenlocher and P. Felzenszwalb. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

D.P. Huttenlocher and W.J. Rucklidge. A multi-resolution technique for comparing images using the hausdorff distance. In *Conference on Computer Vision and Pattern Recognition*, 1993.

Y. LeCun, F. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Conference on Computer Vision and Pattern Recognition*. IEEE Press, 2004.

F. Li, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In *International Conference on Computer Vision*, volume 2, page 1134, 2003.

J.L. Mundy and A. Zisserman, editors. *Geometric Invariance in Computer Vision*. MIT Press, 1992.

B. Ommer, M. Sauter, and J. M. Buhmann. Learning top-down grouping of compositional hierarchies for recognition. In *Conference on Computer Vision and Pattern Recognition*, 2006.

C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, June 2000.

H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–28, 1998.

H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *International Journal of Computer Vision*, 56(3):151–177, 2004.

P. Simard, L. Bottou, P. Haffner, and Y. LeCun. Boxlets: a fast convolution algorithm for neural networks and signal processing. In *Neural Information Processing Systems*, volume 11, 1999.

B. Stenger, A. Thayananthan, P.H.S. Torr, and R. Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28 (9):1372–1384, 2006.

P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

J. Wu, S. C. Brubaker, M. D. Mullin, and J. M. Rehg. Fast asymmetric learning for cascade face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:369–382, 2008.

S.C. Zhu and D. Mumford. *A Stochastic Grammar of Images*, volume 2 of *Foundations and Trends in Computer Graphics and Vision*, pages 259–362. Now Publishers, 2006.