Learning Hyper-Features for Visual Identification

Andras Ferencz **Erik G. Learned-Miller** **Jitendra Malik**
Division of Computer Science
University of California, Berkeley
Berkeley, CA 94720

Abstract

We address the problem of identifying specific instances of a class (cars) from a set of images all belonging to that class. Although we cannot build a model for any particular instance of the class (as we may be provided with only one "training" example of it), we can use information extracted from observing other members of the class. We pose this task as a learning problem, in which the learner is given image pairs, labeled as matching or not, and must discover which image features are most consistent for matching instances and discriminative for mismatches. We explore a patch based representation, where we model the distributions of comparison metrics defined on the patches. Finally, we describe an on-line algorithm that selects the most salient patches based on a mutual information criterion that achieves good performance while only matching a few patches.

1 Introduction

Figure 1 shows six cars: the two leftmost cars were captured by one camera; the right four cars were seen later by another camera from a different angle. The goal is to determine which images, if any, show the *same vehicle*. We call this task *visual identification*. Most existing identification systems are aimed at biometric applications such as identifying fingerprints or faces. While *object recognition* is used loosely for several problems (including this one), we differentiate visual *identification*, where the challenge is distinguishing between visually similar objects of one category (e.g. faces, cars), and *categorization* where the algorithm must group together objects that belong to the same category but may be vi-

Figure 1: *The Identification Problem: Which of these cars are the same?* The two cars on the left, photographed from camera 1, also drive past camera 2. Which of the four images on the right, taken by camera 2, match the cars on the left? Solving this problem will enable applications such as wide area tracking of cars with a sparse set of cameras [2, 8].

Figure 2: *Detecting and warping car images into alignment:* Our identification algorithm assumes that a detection process has found members of the class and approximately aligned them to a canonical view. For our data set, detection is performed by a blob tracker. A projective warp to align the sides is computed by calibrating the pose of the camera to the road and finding the wheels of the vehicle. Note that this is only a rough approximation (the two warped images, center and right, are far from perfectly aligned) that helps to simplify our patch descriptors and positional bookkeeping.

sually diverse[12, 10, 4, 1].[1] Identification is also distinct from "*object localization*," where the goal is locating a specific object in scenes in which distractors have little similarity to the target object [5, 7].

As Figure 1 demonstrates, cars may differ in subtle ways that are dwarfed by appearance changes due to viewing angles and illumination. In such a setting, which image features are informative and which are not? Most previous work in this area uses class-specific features that are hand-selected for the task, such as the distance between the eyes for face identification[9]. Here we present an identification framework that attempts to be more general. The core idea is that while we can not build a model for any particular instance (as we may have only one image of it), we can use information extracted from observing other members of the same class. We pose this task as a learning problem, in which the learner is given image *pairs*, labeled as matching or not, and must discover which image features are most consistent for matching instances and discriminative for mismatches. In testing, novel image pairs must be classified as matched or mismatched.

Our approach learns which features of one image, when compared to the other image, are likely to be informative. A feature is rated by its log likelihood ratio for matching and non-matching pairs, where the likelihoods are conditioned on properties of the feature, such as aspects of its position and appearance. We call these properties *hyper-features*.

The paper is organized as follows. In Section 2, we describe our decision framework including the decomposition of an image pair into *bi-patches*, which give local indications of match or mismatch. In Section 3, we introduce the appearance distance between the two halves as a discriminative statistic of bi-patches. This model is then refined by conditioning the distance distributions on hyper-features such as the patch location, showing improved results. In Section 4, we introduce another comparison statistic, the difference in patch position between images. We extend this positional model to incorporate one type of statistical dependence between bi-patches. Finally, in Section 5, we show a straightforward extension of our method for on-line use. We use a greedy algorithm to compare bi-patches in order of decreasing mutual information with the match/mismatch variable. We show that comparing a small number of well-chosen patches produces performance nearly as good as using all of them.

2 The Decision Rule

We seek to determine whether two different images represent the same vehicle. Let C denote whether two car images match, where $C = 1$ is a match and $C = 0$ a mismatch. Given images I^L and I^R from two cameras (the "Left" and "Right" cameras), and assuming the images are in approximate correspondence (see Figure 2), we pose this task as a decision rule R, where we wish evaluate

$$R = \frac{P(C = 1|I^L, I^R)}{P(C = 0|I^L, I^R)} = \frac{P(I^L, I^R|C = 1)P(C = 1)}{P(I^L, I^R|C = 0)P(C = 0)} > \lambda. \quad (1)$$

[1] There is evidence that this distinction exists in the human visual system. Some findings suggest that the fusiform face area is specialized for identification of instances from familiar categories[11].

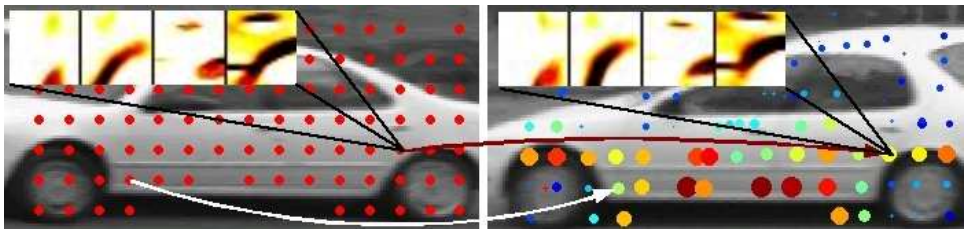

Figure 3: *Patch Matching:* The left image is sampled (red dots) by patches encoded as oriented filter channels (for one patch, four channels are shown). Each patch is matched to the best point in the other image by maximizing the appearance similarity between the patches. For each pair of patches (bi-patch), the size and color of the matching circles indicates similarity in appearance. (Larger and redder is more similar.)

The priors are assumed to be known.[2] λ is chosen to balance the cost of the two types of decision errors, with $\lambda = 1$ indicating they have the same cost. From here forward, for mathematical convenience, we shall calculate $R' = R\frac{P(C=0)}{P(C=1)}$ rather than R. That is, we convert the problem to a setting where the prior probability of a match and a mismatch are equal, thus allowing us to drop the ratio of priors from subsequent equations. The same decision rule is obtained by substituting $\lambda' = \lambda\frac{P(C=0)}{P(C=1)}$ for λ.

To model the likelihoods, we use an image decomposition into patches. In particular, the left image I^L is broken into patches ($F_j^L, j \in \{1...m\}$). For each left patch F_j^L, we find the most similar patch in the right image within some neighborhood around the expected location given by the approximate alignment. We call the combination of a left and a matched right patche (F_j^L, F_j^R), together with their coordinates, a *bi-patch*, denoted F_j. Thus the posterior from Eq.(1) will be approximated as $P(C|I^L, I^R) \approx P(C|F_1, ..., F_m) \propto P(F_1, ..., F_m|C)P(C)$.

For most of this paper, we will assume a naive Bayes model in which, conditioned on C, the bi-patches are assumed to be independent. That is,

$$R' = \frac{P(I^L, I^R|C=1)}{P(I^L, I^R|C=0)} \approx \frac{P(F_1, ..., F_m|C=1)}{P(F_1, ..., F_m|C=0)} = \prod_{j=1}^{m} \frac{P(F_j|C=1)}{P(F_j|C=0)}. \quad (2)$$

In practice, we compute the log of this likelihood ratio, where each patch contributes an additive term (denoted $\mathcal{LLR}_i$ for patch i). Modeling the likelihoods in this ratio is the central focus of this paper.

3 Modeling Appearance Differences

Rather than modeling bi-patch appearances directly, we compute the normalized correlation between an encoding of the left and right image patches.[3] Thus we define

$$d_j = 1 - CorrCoef(F_j^L, F_j^R) \quad (3)$$

to be the "distance" in appearance between the two halves of a bi-patch F_j.

3.1 Baseline Experiment

If we assume that d_j contains all of the information from F_j about the probability of a match, i.e. C and F are independent given d_j, then $P(C|F_j) = P(C|d_j)$. Thus, assuming

[2]For our application, dynamic models of traffic flow can supply the prior on $P(C)$.

[3]Each 25^2 pixel patch is encoded as four oriented filter channels half-wave rectified (where the positive and negative components have been split) for a total of eight channels.

independent bi-patches (and skipping some algebra),

$$R' \approx \prod_{j=1}^{m} \frac{P(F_j|C=1)}{P(F_j|C=0)} = \prod_{j=1}^{m} \frac{P(d_j|C=1)}{P(d_j|C=0)}. \quad (4)$$

In other words, this model assumes that the d_j's are i.i.d. samples from either the $C = 1$ or $C = 0$ distributions.

The two conditional distributions, $P(d_j \,|\, C \in \{0,1\})$, are estimated "non-parametrically" as normalized histograms from training data.[4] For each value of λ, we evaluate Eq. (4) to classify each test pair as matching or not, producing a precision-recall curve. Figure 4 compares this *patch-based* model to a *direct image comparison* method.[5] Notice that even this naive patch-based technique significantly outperforms the global matching.

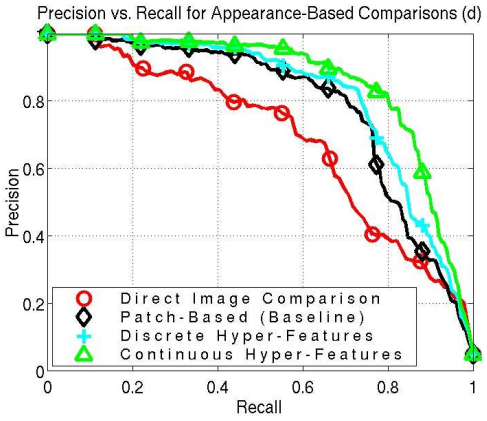

Figure 4: *Identification using appearance differences:* The bottom curve shows the precision-recall curve for non-patch based direct comparison of rectified images. (An ideal precision-recall curve would reach the top right corner.) Notice that all three patch based models outperform this method. The three top curves show results for various models of d_j from Sections 3.1 (Baseline), 3.2 (Discrete), and 3.3 & 3.4 (Continuous). The regression model outperforms the uniform one significantly - it reduces the error in precision by close to 50% for most values of recall below 90%.

3.2 Discrete Hyper-Features

The key observation of our approach is that while the local information for confirming or rejecting a match concerns the *difference* (in both position and appearance) between the left and right patches, the distributions of these differences may depend strongly on the appearance or location of just one of the patches. For example, a large value of d_j computed from a bi-patch in the upper left corner (which is often background) is less informative than the same value near the image center. This observation suggests that we refine our distributions of d_j by conditioning on additional variables such as position. We call these conditioning variables *hyper-features*. We illustrate this idea by conditioning d_j on a simple binary variable Y_j indicating whether the left patch is in the top or bottom half of the image (see *Discrete Hyper-Features* curve in Figure 4). In this model, R' is now approximated as

$$\frac{P(C=1|d_1,Y_1,...,d_m,Y_m)}{P(C=0|d_1,Y_1,...,d_m,Y_m)} = \prod_{j=1}^{m} \frac{P(d_j|Y_j,C=1)P(Y_j|C=1)}{P(d_j|Y_j,C=0)P(Y_j|C=0)} \quad (5)$$

$$= \prod_{j=1}^{m} \frac{P(d_j|Y_j,C=1)}{P(d_j|Y_j,C=0)}, \quad (6)$$

where Eq. (6) follows from the independence of Y_j and C.

[4] Data consisted of 175 pairs (88 training, 87 test pairs) of matching car images from two cameras located on the same side of the street one block apart. Within training and testing sets, about 4000 pairs of mismatched cars were formed from non-corresponding images, one from each camera. All comparisons were performed on grayscale (not color) images.

[5] The global image comparison method used here as a baseline technique uses normalized correlation on a combination of intensity and filter channels, and attempts to overcome slight misalignment.

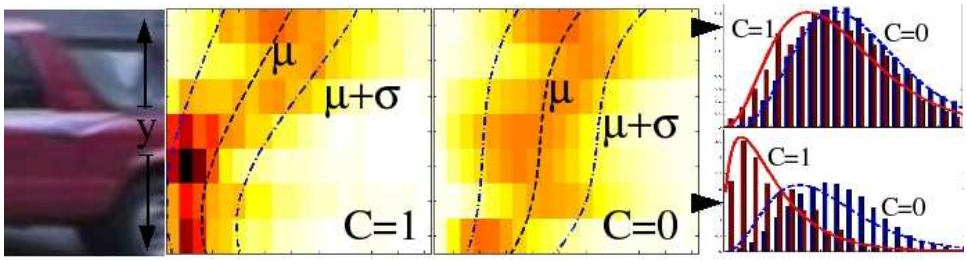

Figure 5: *Fitting a gamma Distribution through μ,σ to the y-position:* we demonstrate our approach by fitting a gamma distribution, through the latent variables $\Theta = (\mu, \sigma)$, to the y position of the patches. Here we allowed μ and σ to be a 3rd degree polynomial function of y (i.e. $\mathbf{Z} = [\mathbf{y^3}, \mathbf{y^2}, \mathbf{y}, \mathbf{1}]^{\mathbf{T}}$). The center-left square shows, on each row, a distribution of d conditioned on the y position of the left patch (F^L) for each bi-patch, for training data taken from matching vehicles. The center-right square shows the same distributions for mismatched data. The height of histogram distributions is color-coded, dark red indicating higher density. The central curve shows the polynomial fit to the conditional means, while the outer curves show the $\pm\sigma$ range. For reference, we include a partial image of a car whose y-coordinate is aligned with the center images. On the right, we show two histogram plots, each corresponding to one row of the center images (a small range of y corresponding to the black arrows). The resulting gamma distributions are superimposed on the histograms, and seem to fit the data well.

3.3 Continuous Hyper-Features

The performance gain from this simple experiment suggests that we may be able to take advantage of more complex relationships between properties of the bi-patches and the distributions over their inter-patch distances. The drawback of introducing additional hyper-features is that the amount of training data available to estimate each conditional distribution is reduced. We mitigate this problem in two ways. First, we assume a parametric form for each conditional distribution, modeling it as a gamma distribution (notated $\Gamma()$) with parameters $\theta = \{\mu, \sigma\}$[6] (Figure 5). This gives each conditional distribution higher bias but lower variance and is appropriate for sparse data conditions.

Second, rather than discretizing the hyper-features into bins and estimating θ separately for each bin, we couple these estimates through four unknown vector hyperparameters $\alpha^{\mu}_{\mathbf{1}}$, $\alpha^{\mu}_{\mathbf{0}}$, $\alpha^{\sigma}_{\mathbf{1}}$, and $\alpha^{\sigma}_{\mathbf{0}}$, where the subscripts refer to the match/mismatch variable C. These hyperparameters are defined as sets of linear weights on the hyper-features and higher-order polynomials of those. Let $\mathbf{X} = \{X_1, ..., X_k\}$ be a set of hyper-features, such as position, contrast, and brightness. Let $\mathbf{Z} = [Z_1, ..., Z_l]^T$ be a vector of various pre-chosen functions of those hyper-features, like squares, cubes, cross terms, or simply copies of the variables themselves. Then each bi-patch distance distribution has the form[7]

$$P(d|X_1, X_2, ..., X_k, C) = \Gamma(d;\ \mu(\alpha^{\mu}_{\mathbf{C}}, \mathbf{Z}),\ \sigma(\alpha^{\sigma}_{\mathbf{C}}, \mathbf{Z})) \quad (7)$$

$$= \Gamma(d;\ {\alpha^{\mu}_{\mathbf{C}}}^T \cdot \mathbf{Z},\ {\alpha^{\sigma}_{\mathbf{C}}}^T \cdot \mathbf{Z}). \quad (8)$$

That is, each distribution is a gamma distribution whose parameters are defined as a polynomial function (i.e. a linear combination of powers) of the conditioning features.[8] This coupling of distributions serves to control the capacity of our model and exploit relationships between the conditional distributions of d to better estimate conditional distributions from sparse data. These ideas are illustrated in Figure 5. The curves are estimated by maximizing the likelihood of the values of d in the training set. Note that the linear weights α^{μ} can be fairly well approximated by a regression fit to d.[9]

[6] The moments can be converted to the natural parameters of the Γ, by $\gamma = (\mu/\sigma)^2$, $\beta = \sigma^2/\mu$.

[7] As neither μ nor σ may be negative, our formula for both includes a minimum value (e.g. $\mu = \max(\mu_0, {\alpha^{\mu}}^T \cdot \mathbf{Z})$) which, for clarity, we have left out of equations.

[8] One could interpret this as approximating $P(d|\mathbf{X}, C) \approx \int P(d|\Theta)P(\Theta|\mathbf{X}, C)d\Theta$ where we have collapsed the integral down to a single maximum likelihood estimate of Θ.

[9] In practice, we use this to initialize our ML optimization of α.

3.4 Automatic Selection of Hyper-Features

In this section we describe the automatic determination of $\mathbf{Z}$, the vector of functions of our conditioning variables. Recall that we assumed a linear relationship between Z and μ, σ. This allows us to use standard feature selection techniques, such as Least Angle Regression (LARS)[3], to choose a small subset of Z_is from a large number of candidates.[10] Specifically, for the experiments reported here, we set $X_1, ..., X_k$ to be: (a) the x and y positions of F^L, (b) the intensity and contrast within F^L and the average intensity of the entire vehicle, and (c) the average energy in each of the 8 oriented filter channels. LARS was then asked to choose $\mathbf{Z}$ from these features in addition to their quadratic, cubic, and cross terms. Once $\mathbf{Z}$ is set, we proceed as in Section 3.3.

Running an automatic feature selection technique on this large set of possible conditioning features gives us a principled method of reducing the complexity of our model. Reducing the complexity is important not only to speed up computation, but also to mitigate the risk of overfitting to the training set. The top curve in Figure 4 shows results when $\mathbf{Z}$ includes the first 10 features found by LARS. Even with such a naive set of features to choose from, the performance of the system improves significantly.

4 Modeling Appearance and Position Differences

In the last section, we only considered the similarity of two matching patches that make up a bi-patch in terms of the *appearance* of the patches (d_j). Recall that for each left patch F_j^L, a matching right patch F_j^R is found by searching for the most similar patch in some large neighborhood around the expected location for the match. In this section, we show how to model the change in *position*, r_j, of the match relative to its expected location, and how this, when combined with the appearance model, improves the matching performance. We start with a model that is very similar to the model for d_j and assumes that the alignment is accurate. Section 4.2 addresses initial misalignment by introducing a global alignment variable that is updated by the matches.

4.1 Change of Position

Let $r_j = (\delta x_j, \delta y_j)$ be the difference in position between the coordinates of F_j^L and F_j^R within the standardized coordinate frames. Generally, we expect $r_j \approx 0$ if the two images portray the same object ($C = 1$). The estimate for R', incorporating the information from both d and r becomes

$$\prod_{j=1}^{m} \frac{P(r_j|d_j, \mathbf{X}_j, C=1)P(d_j|\mathbf{X}_j, C=1)}{P(r_j|d_j, \mathbf{X}_j, C=0)P(d_j|\mathbf{X}_j, C=0)}, \quad (9)$$

where $\mathbf{X}_j$ is again a set of the hyper-features.

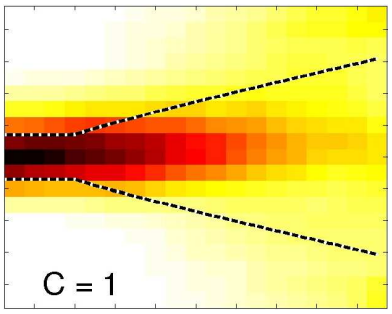

Figure 6: *Distribution of* δy (y axis) conditioned on d (x axis). Superimposed lines show σ_{ML}.

Here we focus on the first factor, where the distribution of r_j given C is dependent on the appearance and position of the left patch (F_j^L, through the hyper-features $\mathbf{X}_j$) and on the similarity in appearance (d_j). The intuition for the dependence on d_j is that for the $C = 1$ case, we expect r_j to be smaller on average when a good appearance match (small d_j) was found (Figure 6).

Following our approach for d_j, we model the distribution of r_j parametrically with a normal distribution with mean 0, $\mathcal{N}(0, \sigma)$, where σ (we use a diagonal covariance) is a function of $\mathbf{X}_j$,d_j. A parameterization of ($\mathbf{X}_j$,d_j) is found through feature selection, while the weights

[10] In order to use LARS (or most other feature selection methods) "out of the box", we use regression based on an $L2$ loss function. While this is not optimal for non-normal data, from experiments we have verified that it is a reasonable approximation for the feature selection step.

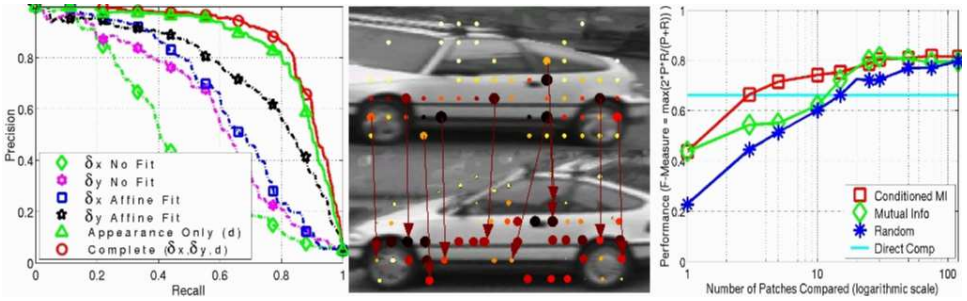

Figure 7: *Results:* The **LEFT** plot shows precision vs. recall curves for models of r. The results for δx and δy are shown separately (as there are often more horizontal than vertical features on cars, δy is better). Re-estimating parameters of the global alignment, W (affine), significantly improves the curves (middle 2 curves). Finally, performance is improved by combining position with appearance (top curve) compared to using appearance alone (2nd from top). The **CENTER** pair of images show a correct match, with the patch centers indicated by circles. The **color of the circles** in the top image = $\mathcal{MI}_j$, in bottom image = $\mathcal{LLR}_j$. Our on-line algorithm (Section 5) chooses patches one at a time based on $\mathcal{CMI}$. The top 10 "left" patches chosen are marked with **arrows** connecting them to the corresponding "right" patches. Notice that these are well distributed in informative regions. The **RIGHT** plot quantifies this observation: the curves show 3 different methods of choosing the order of patches - random order, $\mathcal{MI}$, and $\mathcal{CMI}$. Notice that $\mathcal{CMI}$ with 3 patches does as well as the direct comparison method. All 3 methods converge above 50 patches.

for the linear function are obtained by maximizing the likelihood of r_j over the training data. The bottom two curves in Figure 7 were generated using this method.

4.2 Dependence through Alignment

Until now we have assumed that, conditioned on C, bi-patches are independent (although not identically distributed). While a full treatment of the issues related to dependence within our framework is beyond the scope of this paper, here we briefly note one obvious source of dependence and outline a method for dealing with it. The r_j's are all dependent on the global alignment (W) of I^L and I^R: if W is two pixels off, all r_j's are likely to be, on average, two pixels off. This can be handled in a batch process by estimating a consensus estimate of W (W_e) from all r_j's. This produces an additive term to the distribution of r_j. For example, for the case when W is pure translation, $P(r_j|d_j, \mathbf{X}, C) \approx \mathcal{N}(0, \sigma_j)) + \mathcal{N}(W_e, \sigma_0)$ where σ_0 is the expected standard deviation of the estimated alignment W_e. More complicated models for W, such as affine, can also be supported.

Figure 7 shows that this method (with an affine model) significantly outperforms the version where we assumed that the alignment was correct. While position seems to be less informative than appearance, the complete model (Eq. 9) outperforms appearance alone.

5 On-Line Image Feature Selection Using Mutual Information

Given a set of left, right training examples of other cars, how can we determine, by just looking at the "left" image, which parts, when matched, are likely to be the most informative? It is natural to do this by computing the mutual information between d_j, the distance in appearance between F_j^L and F_j^R and C: $\mathcal{MI}(C, d)$. Given the parametric distributions of d, $P(d|\mathbf{X}, C = 1)$ and $P(d|\mathbf{X}, C = 0)$, as computed in the previous sections, it is straightforward to evaluate $\mathcal{MI}(d, C)$.[11]

This allows us to define an on-line algorithm, which matches patches one at a time, from most informative to least, and stops as soon as a decision can be made. However, a problem arises immediately with this algorithm: nearby patches tend to have similar $\mathcal{MI}$ scores, allowing neighboring patches to be picked successively. What this algorithm ignores is

[11]For computing $\mathcal{MI}$, we set $P(C = 0) = P(C = 1) = 0.5$. Similarly, we could also compute and use $\mathcal{MI}$ between C and the joint distribution of d, r.

the dependence of patches - once one patch in a region has been matched, nearby patches, especially overlapping ones, will provide less information than a patch from a different region. That is, $\mathcal{CMI}(d_1, C|d_2)$, the mutual information of d_1 and C conditioned on an already computed nearby match d_2 should be less than $\mathcal{MI}(d_1, C)$. For lack of space in this paper, we assert without an algorithm that computing a reasonable estimate for $\mathcal{CMI}$ is possible, based on the proximity and similarity of F_1^L and F_2^L. The performance of this patch selection method is compared to pure $\mathcal{MI}$-based sorting and to random selection of the patches in Figure 7. To give a numerical indication of the performance, we note that with only 10 patches, given a 1-to-87 forced choice problem, our $\mathcal{CMI}$-based algorithm chooses correctly 93% of the time.

6 Conclusion

A different approach to a learning problem that is similar to ours can be found in [4], which describes a method for learning object categories from very few training examples of the class. They approach this problem by learning priors on the parameters of a fixed generative model for the category where the training data consists of images from other categories. We, on the other hand, abandon the notion of building a model with a fixed form for an object from a single example. Instead, we take a discriminative approach and model the statistical properties of image patch differences conditioned on properties of the patch. These learned conditional distributions allow us to evaluate, for each feature, the amount of information potentially gained by matching it to the other image.

Our framework is not specific to the features used here but can incorporate a wide variety of feature types, including patch features of different encodings and sizes, as well as global features such as the overall color of the object. By modeling the distributions of comparison metrics on these features using the techniques described in this paper, and applying the mutual information based feature selection, we can build, from a single example, a compact representation in which the features are likely to have high saliency.[12]

References

[1] Y. Amit and D. Geman. A computational model for visual selection. *Neural Computation*, vol. 11, no 7, 1999.

[2] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A Real-time Computer Vision System for Measuring Traffic Parameters. *Proc. CVPR*, 1997.

[3] B. Efron, T. Hastie, I. Johnstone and G. Chu. Least Angle Regression. 2002. http://www-stat.stanford.edu/ hastie/pub.htm

[4] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models for 101 object categories. Computer Vision and Image Understanding, in press.

[5] D. Lowe. Distinctive image features from scale-invariant keypoints. Accepted in *IJCV*, 2004.

[6] E. G. Miller, N. Matsakis, P. A. Viola. Learning from one example through shared densities on transforms. *Proc. CVPR*, 2000.

[7] H. Murase and S. K. Nayar. Learning Object Models from Appearance. *Proc. AAAI*, July 1993.

[8] H. Pasula, S. Russell, M. Ostland, and Y. Ritov. Tracking many objects with many sensors. *Proc. IJCAI*, 1999.

[9] P.J. Phillips, P. Grother, R.J Micheals, D.M. Blackburn, E Tabassi, and J.M. Bone. FRVT 2002: Technical Appendices, March 2003. http://www.frvt.org/FRVT2002/documents.htm

[10] H. Schneiderman, T. Kanade. A Statistical Approach to 3D Object Detection Applied to Faces and Cars. *Proc. CVPR*, 2000.

[11] M.J. Tarr and I. Gauthier. FFA: A flexible fusiform area for subordinate-level visual processing automized by experience. *Nature Neuroscience*, vol. 3, no 8, August 2000.

[12] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proc. CVPR*, 2001.

[12] Answer to Figure 1: top left matches bottom center; bottom left matches bottom right.