Learning Visual Similarity Measures for Comparing Never Seen Objects

Eric Nowak
INRIA - INPG - Bertin Technologies
155 r Louis Armand
13290 Aix en Provence - France
eric.nowak@inrialpes.fr

Frédéric Jurie
LEAR Group - CNRS - INRIA
655 avenue de l'Europe
38334 Saint Ismier Cedex - France
frederic.jurie@inrialpes.fr

Abstract

In this paper we propose and evaluate an algorithm that learns a similarity measure for comparing never seen objects. The measure is learned from pairs of training images labeled "same" or "different". This is far less informative than the commonly used individual image labels (e.g. "car model X"), but it is cheaper to obtain. The proposed algorithm learns the characteristic differences between local descriptors sampled from pairs of "same" and "different" images. These differences are vector quantized by an ensemble of extremely randomized binary trees, and the similarity measure is computed from the quantized differences. The extremely randomized trees are fast to learn, robust due to the redundant information they carry and they have been proved to be very good clusterers. Furthermore, the trees efficiently combine different feature types (SIFT and geometry). We evaluate our innovative similarity measure on four very different datasets and consistantly outperform the state-of-the-art competitive approaches.

Figure 1. Our knowledge of cars allows us to recognize a new car model that we have never seen before, despite changes in pose, light and clutter. This paper proposes an algorithm that performs such a visual identification for never seen objects.

1. Introduction

Humans easily recognize objects even those seen only once. One can recognize a person seen only once, despite changes in dressing, haircut, glasses, expression, etc. One can recognize a car model seen only once, despite changes in pose, light, color, etc (see figure 1). This is because we have a knowledge about our environment, and about persons and cars in particular, thus a single view of a new object of a known category is enough for recognition.

Comparing two images – and more generally comparing two examples – heavily relies on the definition of a good similarity function. Standard functions (e.g. the Euclidean distance in the original feature space) are often too generic and fail to encode *domain specific knowledge*; this is why we propose to learn a similarity measure that embeds domain specific knowledge.

Moreover, we propose to learn this measure from *equivalence constraints*. Equivalence constraints considered in this paper are pairs of training examples representing similar or different objects. A pair of images is not labeled "car model X and car model Y", but only "same" or "different". The later is much more difficult because it contains less information: same or different pairs can be produced from fully labeled examples, not vice versa. For many applications, equivalence information is cheaper to obtain than labels, e.g. for retrieval systems. It is indeed easier to know whether two documents are similar or not rather than to obtain their true labels, because the space of potential labels is very large (e.g. all car models) and difficult to define.

We use this similarity measure for *visual identification of never seen objects*. Given a training set of pairs labeled "same" or "different", we have to decide if two never seen objects are the same or not (see figure 2).

1-4244-1180-7/07/$25.00 ©2007 IEEE

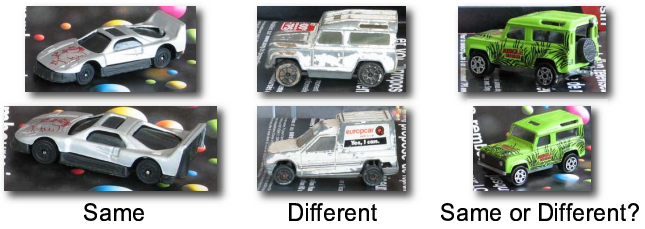

Figure 2. Given pairs labeled "same" or "different", can we learn a similarity measure that decides if two images represent the same object? The similarity measure should be robust to modifications in pose, background and lighting conditions, and above all should deal with *never seen* objects.

1.1. Related Works

Learning effective functions to compare examples is an active topic which received much attention during the last years. Most of the contributions consist in finding a function mapping the feature space into a target space such that a simple distance can eventually be used in the target space.

This function is generally inspired by the Mahanalobis distance, of the form $d(x, y) = (x - y)^t A(x - y)$, like in [14] or more recently [21, 11, 19, 1, 10, 20]. Various optimization schemes are possible to estimate A, depending on the objective function to be satisfied. The objective function plays a key role in the definition of the metric. In [21, 11] the objective function tries to collapse all examples of the same class and to separate examples of different classes. In [11], a stochastic variant of the leave-one-out k-NN score is maximized. In [20], the objective function tries to separate examples from different classes by a large margin, in a k-NN framework. In [19], the margin between positive pairs and negative pairs is to be maximized. In [1], A is directly computed from the so-called *chunklets*, which are the sets of equivalence relations provided as training data. The mapping can also be learned without explicit functions, like in [3] where a convolutional network is used for its robustness to geometric distortions. When considering distance between images, more specific functions can be used, embedding expected deformations of object appearances [16, 6, 13].

Unfortunately, none of these methods is perfectly suited for visual identification in images. Unlike traditional pattern recognition problems, information included in images is subject to complex transformations such as occlusions, pose and scale changes, etc., that can not be modeled easily by any kind of linear, quadratic, or other polynomial transformations.

The usual way to face these problems is to represent images as a collection of lose scale invariant local information (gray-scale patches, SIFT [15] descriptors or others), so that at least several parts of the image are not affected by these transformations. This kind of strategy has been used in [4, 5]; in this case, the key idea is to learn what characterizes features (local descriptors) that are informative in distinguishing one object instance from another. There is also an approach based on *chopping* [7]. However, this approach, which relies on random binary splits chosen to keep images of the same object together, requires to have all the training images fully labeled and therefore it is not usable in our context. Finally, the recent approach of Frome et al. [8] learns a distance function for each individual training image as a combination of elementary distances between visual features. Their approach is based on triplets of images (F, I_1, I_2) with F more similar to I_1 than I_2.

Inspired by the work proposed in these related approaches and more particularly in [5], we propose a new learning method for measuring similarity between two images of never seen objects, using information extracted from pairs of similar and different objects of the same generic category.

Our approach is also inspired by the recent work of [17]. Several key components are responsible for its good performance. First, a *bag-of-words* like model makes it robust to occlusions and various image transformations; second, the use of an ensemble of extremely-randomized trees makes it very fast (both for training and testing) and gives good properties when dealing with high-dimensional features (image patches).

The paper is organized as follows. In section 2 we present our approach and in section 3 we show experimental results obtained on different datasets. We also compare our results with those obtained by several recent competing approaches (section 3.3).

2. Building a similarity measure from patch correspondences

As explained earlier, our objective is to build a similarity measure for deciding whether two images represent the same object instance or not, despite view point changes, occlusions and other image transformations (see figure 2). This measure is expected to give good results when the objects involved in the comparison have never been seen before. Furthermore, the system is designed to be trained from pairs of "same" and "different" objects, without knowing their labels: we assume having no information about which objects are in the training pairs.

2.1. Quantizing local differences

As in [5], we propose to observe corresponding local regions sampled from pairs of images, but we do not limit our observation to the distance between the region descriptors, we also want to describe how the regions differ. Thus, we propose to *characterize the difference* in appearance of corresponding local regions. This is achieved by clustering the

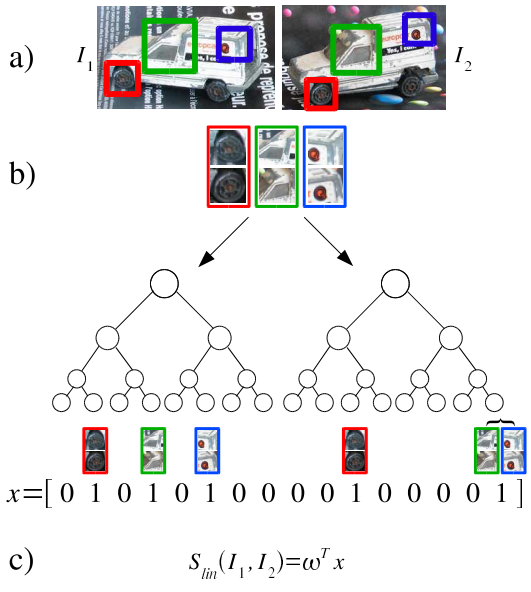

Figure 3. Similarity computation. (a) Detect corresponding patch pairs. (b) Quantize them, i.e. assign them to clusters via extremely randomized trees. (c) The similarity is a linear combination of the cluster memberships.

local differences with an ensemble of extremely randomized trees, i.e. by building a visual dictionary of corresponding patch differences.

Let us consider how to characterize two almost similar local regions of two different objects (e.g. a wheel of two cars). We may quantize each local descriptor and then measure how the quantized descriptors differ. We may also compute the difference between the local descriptors and then quantize that difference. The former computes the precise difference of two approximate (because quantized) local regions, whereas the latter computes the approximate difference of two precise local regions. The former is not precise, because once regions are quantized, fine differences are lost, whereas the latter is able to represent fine differences. This is the reason why we are not dealing with quantized local regions but with *quantized difference* of local regions, as it describes the structure of the space of local region differences while preserving fine differences.

2.2. Overview

The computation of the similarity measure is a three step process illustrated on figure 3. (a) Several pairs of corresponding local regions (patches) are sampled from a pair of images. (b) Each patch pair is quantized, i.e. it is assigned to several clusters with an ensemble of extremely randomized decision trees. (c) The cluster memberships are combined to make a global decision about the pair of images. These steps are detailed below.

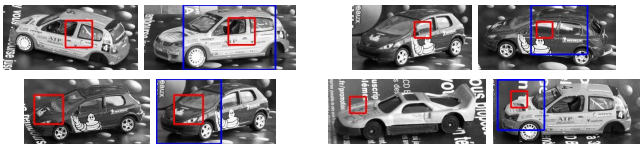

Figure 4. Patch pairs sampled on a toy car dataset. Each image pair shows: a random patch in image 1, the search region in image 2 and the best match in the search region. All image pairs are positive (same object) except the last one (different objects).

2.3. Computing the similarity of two images

Sampling corresponding patch pairs. Each patch pair is produced as follows. A patch p_1 of a random size is chosen at a random position (x, y) in the first image I_1. The best normalized cross correlation match p_2 is looked for in the second image I_2, in the neighborhood of (x, y). p_1 and p_2 are then resized to a standard size w_{std} so that all patches can be compared, independently of their scale. The process is illustrated on figure 4.

Quantizing the space of patch pairs. Each patch pair sampled from an image pair is assigned to several clusters via an ensemble of extremely randomized binary decision trees. How we build them is detailed in section 2.4. Each patch pair is input in the root node of all trees (see figure 3). For each tree, the patch pair goes from the root node to a leaf, at each node the left or right child node is selected according to the evaluation of a simple test on the patch pair. When a patch pair reaches a leaf, the corresponding leaf label (i.e. the id of the leaf in the forest) is set to 1. If a leaf is never reached, it is set to 0. Thus, an image pair is transformed into a binary vector x (of size the total number of leaves), each dimension indicating if a patch pair sampled from the image pair has reached the corresponding leaf. The use of binary representation to indicate cluster membership has been suggested by [17].

The similarity measure. The learned trees perfectly discriminate the patch pairs they were trained on since they were trained to do so. However, we are not using the trees as classifiers, but as quantizers, so we consider which leaves are reached by the patch pairs and discard the prediction of the decision trees.

The similarity measure is a simple linear combination of the binary feature vector indicating cluster membership: $S_{lin}(I_1, I_2) = \omega^\top x$, where ω contains weights optimized such that high values of S_{lin} correspond to similar images. In practice, ω is the hyperplane normal of a binary linear SVM trained on positive and negative image pair representations.

The weights are a very convenient way to express how useful a leaf (i.e. cluster) is. Intuitively, a leaf which is

equally probable in positive and negative image pairs should not weight much because it is not informative. On the contrary, a leaf which occurs only in positive or negative image pairs should weight more.

2.4. Learning the extremely randomized trees

All trees are learned independently, according to a procedure suggested by Geurts [9] which is a Perturb and Combine paradigm [2] applied to decision trees. We sample a large number of patch pairs from positive image pairs (same object) and negative image pairs (different objects). We then create a tree with a unique node, the root node, that contains these positive and negative patch pairs. We recursively split the nodes and their associated patch pairs to create a tree: we assign a random boolean split condition to a node, create two sub-nodes, one with patches for which the condition is true, the other one with the remaining patches; we repeat this procedure as long as the sub-nodes contain positive *and* negative patch pairs.

The boolean split conditions (detailed in the next section) are parametrized functions evaluated on the patch pair. We generate a small set of split conditions with random parameters, and keep the one with the highest information gain: $IG = H - (n_1 H_1 + n_2 H_2)/n$ where H (resp. H_1, H_2) and n (resp. n_1, n_2) are the entropy and the number of patches of the parent (resp. first child, second child).

We are using a set of extremely randomized binary decision trees for several reasons. First, learning the trees is fast because unlike boosting or ID3, we are not looking for optimal parameter values, we are only selecting the best ones out of a randomly created small set. Second, the use of several extremely randomized trees decreases the risk of overfitting, because it makes the trees less correlated [2]. Third, such trees have been proved to provide an interesting clustering of the feature space [17].

2.5. Multi-modal split-conditions

We propose and combine two different kinds of split conditions. The first kind uses pixel information, the second one uses geometry information. For the former, we considered graylevel pixel values, gradient norm and orientation, and SIFT descriptors. As SIFT descriptors always outperform the other ones, we only focus on SIFT descriptors in this paper.

SIFT based split-conditions. Given the SIFT descriptors S_1 and S_2 of two patches, the split condition is true if

$$k(S_1(i) - d) > 0 \ \wedge \ k(S_2(i) - d) > 0 \qquad (1)$$

where i, d, k are parameters. i is the SIFT dimension under observation, d is a threshold, and $k = 1$ or $k = -1$ encodes if the measured value should be higher or lower than the threshold.

Geometry based split-conditions. Given the position and scale (x, y, s) of the patch from the first image, the split condition is true if

$$k_x(x - d_x) > 0 \ \wedge \ k_y(y - d_y) > 0 \ \wedge \ k_s(s - d_s) > 0 \quad (2)$$

where $d_x, d_y, d_s, k_x, k_y, k_s$ are parameters. k_x, k_y, k_s are equal to 1 or -1 and encode if the values should be above or below the thresholds d_x, d_y, d_s. This split condition can encode complex concepts, for example a large patch in the bottom left corner of an image may be

$$-1{*}(x{-}0.25) > 0 \wedge 1{*}(y{-}0.75) > 0 \wedge 1{*}(s{-}0.5) > 0$$

For each tree node, we generate random boolean tests of any type (SIFT/geometry): first we randomly draw a type, then we randomly draw the parameters it requires. The information gain is computed for all these boolean tests, and the best one is assigned to the node.

3. Experimental results

We evaluate our similarity measure on four different datasets: a small dataset of toy cars and three other publicly available datasets, making comparisons with competitive approaches possible. For each dataset, the objects of interest fully occupy the images and we have pairs marked as positive (same object) or negative (different objects). Those sets are split into a training set and a test set. Obviously, the test set does not contain any image from the training set, but it does not contain any object of the training set either. The similarity measure is evaluated only on *never seen* objects (not only never seen images). The datasets are detailed below, and they are illustrated on figure 5.

The toy cars dataset[1] contains 225 images of 14 different objects (cars and trucks). The training set contains 1185 positive and 7330 negative image pairs of 7 different objects. The test set contains 1044 positive and 6337 negative image pairs of the 7 other (new) objects.

The Ferencz & Malik cars dataset [5] contains 2868 training pairs (180 positive, 2688 negative) and the test set contains 2860 pairs.

The Jain faces dataset [12] is a subset of "Faces in the news"[2] and contains 500 positive and 500 negative pairs of faces, and we measure our accuracy like the authors by 10 fold cross validation. That dataset is built from faces sampled "in the news", hence there are very large differences of resolution, light, appearance, expression, pose, noise, etc.

The Coil-100 dataset used by Fleuret & Blanchard [7] has 10 different configurations, each configuration uses 1000 positive and 1000 negative pairs from 80 objects for training and 250 positive and 250 negative pairs from the remaining 20 (new) objects for testing. This dataset is highly

[1] http://lear.inrialpes.fr/people/nowak/dwl/toycarlear.tar.gz

[2] We thank the authors for providing us the precise subset

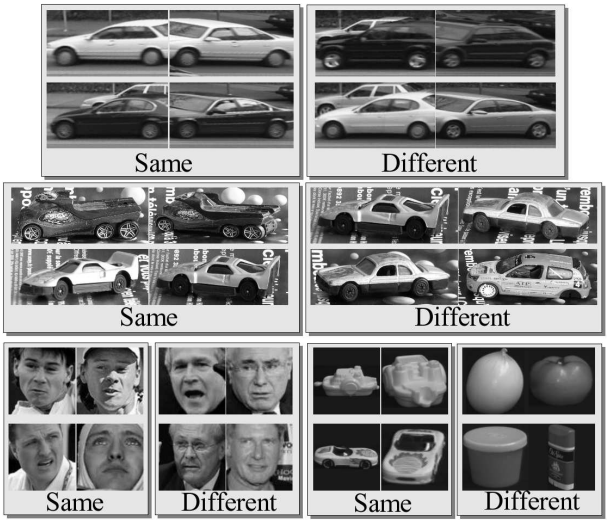

Figure 5. Two "Same" and two "Different" pairs from all datasets. Line 1: Ferencz cars, Line 2: our toy cars, Line 3 left: Faces in the News, Line 3 right: Coil 100. Although "Different" pairs may look similar and "Same" pairs may look different and test set objects are never seen, our similarity measure obtains a very high performance on all these datasets (see section 3.3).

heterogeneous, as it contains categories such as bins, tomatoes, boxes, medicine, puppets, mugs, bottles, ...

In all experiments, gray-scale and color images are all considered gray-scale. All datasets have image pairs of slightly different orientations, except Coil-100 that has very different orientations.

To evaluate the performance, we compute a Precision-Recall Equal Error Rate (EER PR) score on the similarity measure evaluated on the test set image pairs. For each test set image pair, a similarity score S_{lin} is computed. A threshold t is defined to decide if the two images represent the same object instance or not: $S_{lin} > t$ means "same" object, $S_{lin} \leq t$ means "different" objects. The Precision-Recall curve is obtained by varying the threshold t.

3.1. Parametric evaluation

We have evaluated the influence of all the parameters involved in the similarity measure using the toy car dataset. Figure 6 shows the effect of influential parameters. Each experiment plots the Precision-Recall Equal Error Rate (EER-PR) w.r.t. a parameter. We give the EER-PR of our similarity measure S_{lin} as well as the EER-PR of a simple vote based similarity measure S_{vote} that uses the trees as classifiers and not clusterers, and thus counts the number of patch pairs predicted as "same".

We first notice that the linear similarity measure S_{lin} always outperforms the simple similarity S_{vote}, which proves that trees are more useful as clusterers instead of classifiers. Second, let us discuss the different parameters one

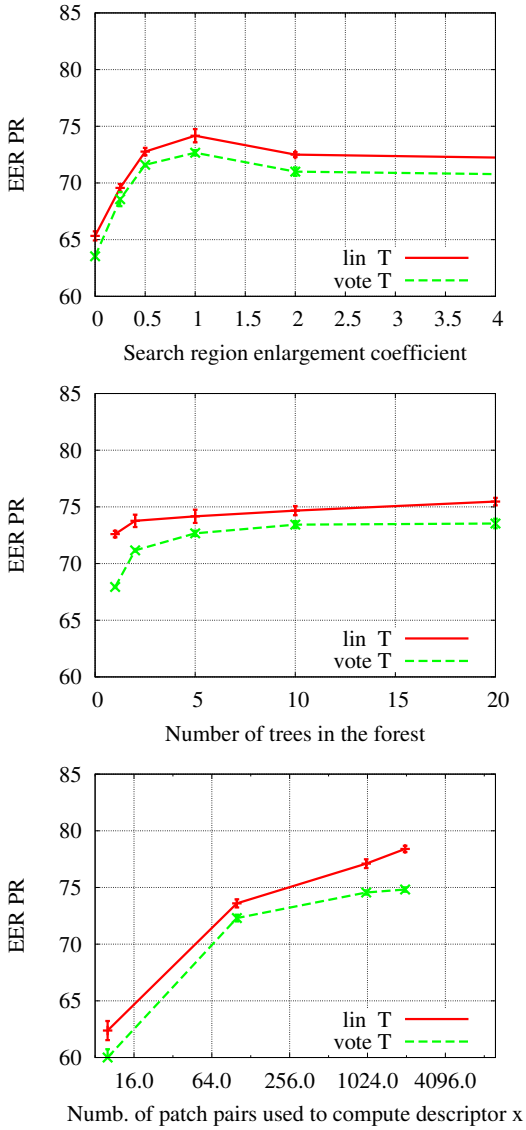

Figure 6. Precision-Recall Equal Error Rate for the toy car dataset, with a simple similarity measure S_{vote} and our linear similarity measure S_{lin}.

by one. The first curve shows that the search region size should not be too small (otherwise it is impossible to find the expected match) nor too large (leading to too many misleading matches). Moreover, if the second patch is selected randomly, the performance is very bad (42.7%, not shown on the graph). This shows how crucial the computation of good patch pairs is. The second curve shows that the more trees, the higher the performance. This is because we obtain more clusters, and because the clusters are not correlated due to the randomness of the tree computation. The third curve shows that the more patch pairs sampled in an image pair the higher the performance. We first believed

	Toy cars		Ferencz		Faces		Coil 100	
	T+W	T	T+W	T	T+W	T	T+W	T
Ferencz	28.0	4.5	0	0	49.2	23.5	35.8	10.1
Coil100	10.0	1.6	9.0	2.4	13.2	2.8	0	0

Table 1. Trees (T) and weights (W) are learned by our algorithm. This table shows the decrease of EER-PR evaluated on the Ferencz and Coil100 datasets when the trees (T) or the trees and the weights (T+W) are learned on other datasets.

that sampling more windows increases chances to sample *relevant* information. But if it were true, only S_{lin} would progress because it is able to separate relevant and irrelevant information. However, S_{vote} also increases, and that measure makes no difference between relevant and irrelevant information. This means that any "weak" information also improves the performance, which confirms our previous works [18] about sampling strategies.

Also, on average, using SIFT and geometry is 1% better than using SIFT only. The increase is surprisingly low as some categories have strong geometric structures. But this is not negligible.

3.2. Generic versus specific knowledge

Our algorithm learns two types of information from the training data: the trees and the weights of the similarity measure. It is interesting to investigate how much *category specific knowledge* is embedded during training. Does the algorithm learn generic rules, or does it use heuristics specific to the dataset it is trained on? Elements of the answer are presented in Table 1. It shows, for two datasets, how the EER-PR decreases when other datasets are used for learning. When testing on a dataset, another dataset may be used to learn the trees and the weights (T+W), or the trees only (T), in which case the weights are learned on the training set of the good dataset.

First, we observe that the best performance is achieved when training and test are performed on the same dataset, which means that category specific knowledge is embedded during learning. Second, learning the trees and the weights on another dataset is always much worse than learning the trees only. It means that the weights allow to use any clusterer, although the performance is better when the best clusterer is used. Third, it is more important to use the appropriate dataset to learn the trees for the Ferencz dataset than for the Coil 100 dataset. This is because Coil 100 images may have any orientation and any shape, and they represent very different objects, whereas the Ferencz dataset contains aligned images of cars seen from profile, which makes it easier to learn very specific (and useful) information.

Method	Toy cars	Ferencz	Faces	Coil 100
Others	-	84.9 [4]	70.0 [12]	$88.6_{\pm 4}$ [7]
Ours	$85.9_{\pm 0.4}$	$91.0_{\pm 0.6}$	$84.2_{\pm 3.1}$	$93.0_{\pm 1.9}$
Gain	-	6.1	14.2	4.4

Table 2. PR-EER on different datasets. Our method clearly outperforms the others.

Method	Recall 40%	Recall 60%	Recall 80%
Jain [12]	$93.0_{\pm 6.3}$	$78.9_{\pm 8.2}$	$60.1_{\pm 7.0}$
Ours	$99.0_{\pm 1.9}$	$97.8_{\pm 2.5}$	$86.3_{\pm 7.6}$

Table 3. Precision on the Jain faces dataset for given recall.

3.3. Performance and comparison with state-of-the-art competitive approaches

We report the performance of our algorithm and compare our results with state of the art methods on the four datasets. For each dataset, the experiments are carefully performed with the same protocol as the one used in the method compared to. All the results are summarized in Table 2.

For all these experiments on the different datasets, we use the same parameters: the number of positive and negative patch pairs sampled to learn the trees is set to 10^5, the number of random split conditions among which the best one is selected is 10^3, the number of trees in the forest is 50, the number of patch pairs sampled to compute the similarity is 10^3, the second patch search region size is increased of 1 time the size of the patch in all directions, leading to a search region 9 times larger than the original patch, the minimum patch sampling size is set to 15x15 pixels, and the maximum is set to one half of the image height. On average, the produced trees have 20 levels.

Toy car dataset. On our toy car dataset, we obtain a precision-recall equal error rate (EER PR) of 85.9%$\pm$0.4 measured on 5 runs. Using a simple Normalized Cross-Correlation (NCC) as a similarity measure leads to an EER PR of 51.1%. This is a new dataset for which no other results are published yet.

Ferencz & Malik dataset. On the Ferencz & Malik dataset, we obtain an EER-precision of 91.0%$\pm$0.6, where Ferencz & Malik [4] get 84.9%. Figure 7 shows pairs of cars ranked by similarity.

Faces in the news dataset. Jain [12] greatly outperforms the top performer in the FERET face recognition competition on their face dataset, and we largely outperform their results. The EER-PR reported in Table 2 (70%) is approximate since we have estimated it from a curve in their paper, thus we also provide a comparison with the same metric as they use, the precision score for given recall values, see Table 3. We always outperform their results, and are even 26% better for a 80% recall. Moreover, Jain is working on face images rectified to frontal pose, whereas we are working

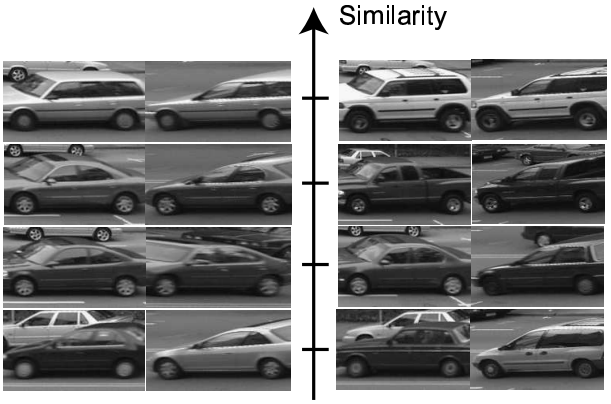

Figure 7. Test set image pairs from Ferencz & Malik dataset, line 1: the two most similar pairs, line 2: the two least similar pairs, line 3: the two least different pairs, line 4: the two most different pairs (according to the learned similarity measure).

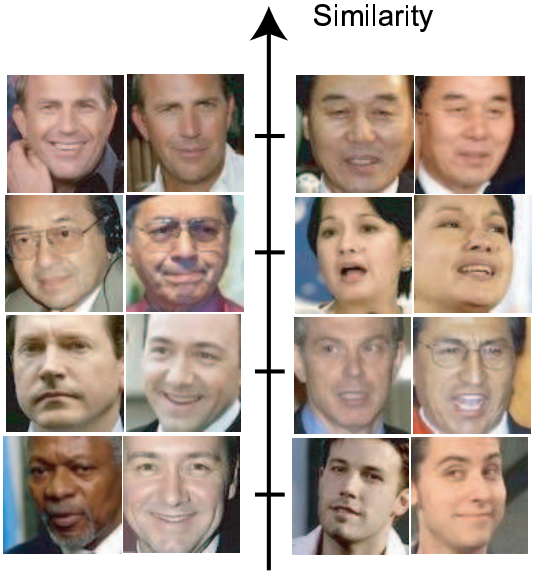

Figure 8. Test set face pairs ranked by our similarity measure: line 1: two very similar pairs, line 2: two similar pairs, line 3: two different pairs, line 4: two very different pairs.

on the original face images. Figure 8 shows pairs of faces ranked by similarity.

Coil-100 dataset. On the Coil-100 dataset, we have an EER-PR of 93.0%±1.9 where Fleuret & Blanchard [7] have 88.6%±4. Moreover, the method of Fleuret and Blanchard uses the information of the real object categories during training, whereas we only know if two images belong to the same category or not.

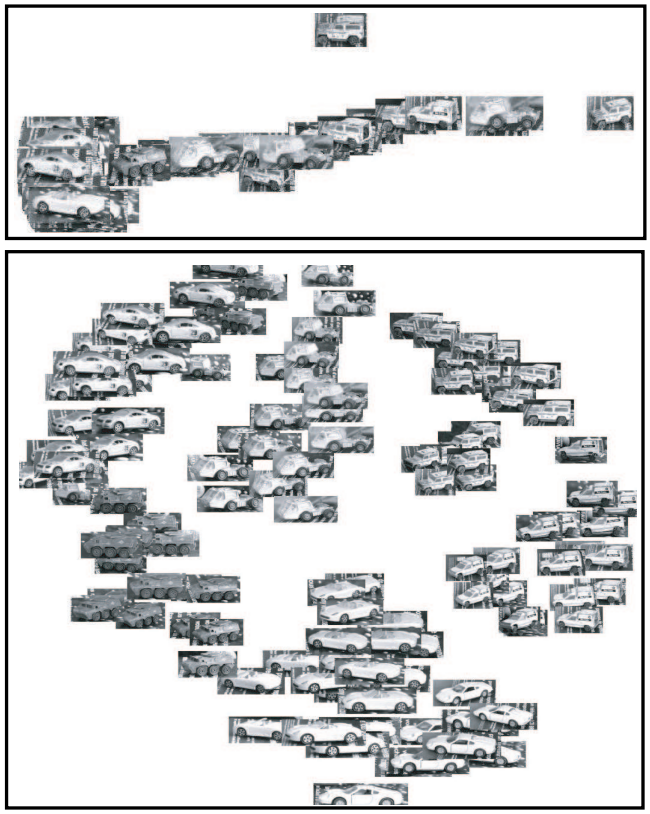

Figure 9. 2D multidimensional scaling representation of never seen toy car images. Top: similarity measure based on bag-of-words representation and Euclidean distance. Bottom: our similarity measure, that perfectly groups the different views of the same object.

3.4. Visualizing the similarities

The previous section shows that our similarity measure outperforms the ones that achieve the best state of the art performance. Since "a picture is worth a thousand words", we also compute a 2D visualization of the similarity measures evaluated on the never seen images of the toy car dataset.

Therefore we compute a 2D mapping of images using the learned similarity function by applying a multidimensional scaling technique (MDS). It computes the 2D projection preserving as much as possible all pairwise image similarities (Sammon mapping). This mapping can be seen on Figure 9, bottom. It is surprising to see how well different views of same objects are grouped together despite large intra-class variations, high inter-class similarities and despite the fact that these cars are very different from the ones used to learn the distance. The top of the figure shows the same mapping using only bag-of-word representations of images and the Euclidean distance.

Figure 10. All positive patch pairs contained in a node during tree learning on the face dataset.

4. Discussion and Conclusion

We are addressing the problem of predicting how similar two images of never seen objects are, given a set of similar and different training object pairs. We propose a novel method consisting in (a) finding corresponding local regions in a pair of images (b) quantizing (clustering) them with an ensemble of extremely randomized trees and (c) combining the cluster memberships of the local region pairs to compute a global similarity measure between the two images.

Our algorithm automatically selects and combines geometry and SIFT features. We have shown that it does not perform a generic similarity computation, but that it embeds knowledge specific information, via the construction of the trees and the computation of the optimal weights.

Our experiments show that our approach gives excellent results on the four datasets used for evaluation. We greatly outperform the latest results of Ferencz *et al*. [4], Vidit *et al*. [12] and Fleuret *et al*. [7] significantly, and obtain a high accuracy on our own dataset.

We are currently extending our approach to recognize similar *object categories* from a training set of equivalence constraints. Positive image pairs may contain two different models of bikes, cars, motorbikes, etc. This is more difficult than dealing with positive pairs of the same model, because it is harder to obtain two local regions that match from images of two different models, moreover if the objects may have any orientation and scale.

References

[1] A. Bar Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, (6):937–965, 2005.

[2] L. Breiman. Random forests. *ML Journal*, 45(1):5–32, 2001.

[3] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR '05*, pages 539–546, 2005.

[4] A. Ferencz, E. Learned Miller, and J. Malik. Building a classification cascade for visual identification from one example. In *ICCV'05*, pages I: 286–293, 2005.

[5] A. Ferencz, E. Learned-Miller, and J. Malik. Learning hyperfeatures for visual identification. In *NIPS'05*, pages 425–432. 2005.

[6] A. Fitzgibbon and A. Zisserman. Joint manifold distance: a new approach to appearance based clustering. In *CVPR'03*, pages I: 26–33, 2003.

[7] F. Fleuret and G. Blanchard. Pattern recognition from one example by chopping. In *NIPS'05*, pages 371–378. MIT Press, 2005.

[8] A. Frome, Y. Singer, and J. Malik. Image retrieval and classification using local distance functions. In *NIPS'06*, 2006.

[9] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 36(1):3–42, 2006.

[10] A. Globerson and S. Roweis. Metric learning by collapsing classes. In *NIPS'05*, 2005.

[11] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *NIPS'04*, 2004.

[12] V. Jain, A. Ferencz, and E. Learned Miller. Discriminative training of hyper-feature models for object identification. In *BMVC'06*, pages I, pp. 357–366, 2006.

[13] F. li, R. Fergus, and P. Perona. One-shot learning of object categories. *PAMI*, 28(4):594–611, April 2006.

[14] D. Lowe. Similarity metric learning for a variable-kernel classifier. *Neural computation*, 7(1):72–85, 1995.

[15] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, November 2004.

[16] E. Miller, N. Matsakis, and P. Viola. Learning from one example through shared densities on transforms. In *CVPR'00*, pages I: 464–471, 2000.

[17] F. Moosmann, B. Triggs, and F. Jurie. Randomized clustering forests for building fast and discriminative visual vocabularies. In *NIPS'06*. 2006.

[18] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *European Conference on Computer Vision*. Springer, 2006.

[19] S. Shalev-Shwartz, Y. Singer, and A. Y. Ng. Online and batch learning of pseudo-metrics. In *ICML '04*, 2004.

[20] K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS'05*. 2006.

[21] E. Xing, A. Ng, and S. Jordan M.I.and Russell. Distance metric learning, with application to clustering with side-information. In *NIPS'02*, 2002.