Learning a discriminative classifier using shape context distances

Hao Zhang Jitendra Malik
Computer Science Division
University of California at Berkeley
Berkeley, CA 94720-1776

Abstract

For purpose of object recognition, we learn one discriminative classifier based on one prototype, using shape context distances as the feature vector. From multiple prototypes, the outputs of the classifiers are combined using the method called "error correcting output codes". The overall classifier is tested on benchmark dataset and is shown to outperform existing methods with far fewer prototypes.

1. Introduction

The problem of visual object recognition has posed a number of challenges to existing vision and learning techniques. One of the most significant difficulties is the high visual variability of objects in the same class. In 3-D, pose and illumination changes can cause drastic changes in appearance of the object. Even 2-D objects such as letters and digits have high intra-class variation. For example, many alphabets have "allographs", which are different ways to write the same letter. Human vision system is naturally adapted to such variation among the visual objects: cognition is invariant under certain types of transformation. However, it is not obvious how to build such invariance into current machine vision systems.

Despite the lack of a perfect representation of visual objects, supervised learning has provided a natural and successful framework for studying object recognition. Working with color, texture and shape cues, a number of learning techniques have shown success in their respective problem domains. Among these approaches, some make careful choices of the representation of the object, so as to obtain a rich and meaningful feature descriptor; others put more emphasis on the "learning" aspect of the recognition task, making use of more sophisticated learning techniques. It is the aim of this paper to try to combine the best of both worlds.

1.1. Shape matching

Many methods on shape matching can be found in the survey of [19]. They are roughly divided into brightness-based or feature-based methods. Brightness-based methods work with the intensity of the pixels and use the image itself as a feature descriptor. Feature-based methods extract points from the image (usually edge or corner points) and reduce the problem to point set matching. Some early work aimed to capture the structure of the shape using the medial axis transform, e.g. [17]. A class of methods working with silhouettes establish correspondence between contours of shapes, e.g. [12]. Working with edges from an image, [9] developed methods using Hausdorff distance. Methods without recovering correspondences are also used to measure the similarity between shapes, for example, geometric hashing [11] uses configuration of keypoints to vote for a particular shape. Recent work on shape context [3] has shown its promise as a feature descriptor that works well with learning techniques such as nearest neighbor.

1.2. Learning

In the area of face recognition, the use of PCA has produced good results [18]. On more general object recognition tasks, several other learning methods such as Bayes classifier [16] and decision tree learning [2] have been applied. The technique of boosting is shown to be a viable way of feature selection in [20]. Support vector machine methods have shown success in template matching problems (recognizing pedestrians) [14]. Indeed, a broad class of object recognition problems have benefited from statistical learning machinery.

Another important example is digit classification as applied with neural network in [13]. The digit data set used in that paper (MNIST) has been tested on many different pattern recognition algorithms and was also included in [3], in which the lowest error rate was obtained.

The recent technique of error-correcting output codes (ECOC) [6] has shown to be a successful alternative to nearest neighbor in the problem of multiway classification. It generalizes the conventional wisdom of "one-against-all" method, and other more sophisticated methods such as pairwise coupling[7]. In [1], the ECOC method, combined with boosting, was applied, with good success, to the digit and letter subsets of the UCI dataset. The UCI dataset provides features that are already extracted from the images. It is interesting to see how the ECOC method would work with real images and whether it would benefit from rich feature descriptor such as shape context.

1.3. Overview

The shape of an object is not readily described by a feature vector in $\mathbb{R}^n$, which is the basis of many learning techniques. In this work, we let go of the global sense of a feature vector. We anchor on prototype shapes and try to define a *local* feature vector based on that prototype, using shape context distances reviewed in section 2. In this restricted setting, we maximize the discriminative power of the classifier through learning procedures. The details are described in section 3.

The outputs from the local classifiers are then combined using error-correcting output codes, which gives a substantial improvement over the naive procedure of taking the maximum response (which amounts to nearest neighbor). This is the subject of section 4.

We test our method on the MNIST dataset, a basis for comparison among many learning algorithms. It should be noted that our technique can be extended to general, multi-class object recognition tasks with almost no change. With that said, we present the results on digit classification in section 5.

2. Previous work on shape context

In [3], the shape context has been shown to be a powerful tool for object recognition tasks, including classification from binary images and from images of 3D objects under various poses.

The basic idea of shape context is illustrated in Fig. 1(a). The shape of an object is represented as a discrete set of points on the contour. A subset of the contour points is selected as centers for computing the shape context, defined as a log-polar histogram. The count in each bin is drawn from all of the contour points, providing a description of the entire shape relative to the center.

Given two shape contexts, which are histograms, one can measure how likely they come from the same underlying distribution. Standard statistical methods such as χ^2-test can be used.

In this work, we use the χ^2-distance defined as:

Let $H_X(1..n)$ and $H_Y(1..n)$ be the bin counts of two shape contexts. Normalize them to obtain empirical density function h_X and h_Y: $h_X(i) = \frac{H_X(i)}{\sum_{i=1}^n H_X(i)}$, $h_Y(i) = \frac{H_Y(i)}{\sum_{i=1}^n H_Y(i)}$. The similarity measure between the two shape contexts is then defined as:

$$\chi^2(H_X, H_Y) = \frac{1}{2}\sum_{i=1}^{n} \frac{|h_X(i) - h_Y(i)|^2}{h_X(i) + h_Y(i)}$$

Note that from a property of the χ^2 test, this similarity measure is between 0 and 1. A small value of the χ^2 statistic tells us that the two shape contexts are likely to come from corresponding points on two shapes, a situation illustrated in Fig. 1. From the distance measure between pairs of shape contexts, one can guess which points from either shape are in correspondence. However, not all of the guesses are correct. [3] proposed an iterative procedure to improve the matching: given an initial guess of correspondence between shape A and shape B, estimate the transformation that brings A into B, then apply the transformation to A, obtain A' and repeat the process on A' and B, until the two shapes are sufficiently close. To estimate the transformation, they restrict their attention to the family of Thin Plate Splines [4], an extension of minimal-energy splines of one variable into 2 dimensions. An example of the matching process is shown in Fig. 2(a).

Between shapes A and B, the SC+TPS matching procedure returns three types of distance measures: D_{sc}, D_{ac} and D_{be}. The D_{sc} term measures the distance between a point on shape A and its closest point on shape B,in the sense of shape context distance (the most "look-alike" point). The D_{ac} term is computed for each point on shape A the sum of squared difference in gray levels of a surrounding patch with its corresponding point on shape B, illustrated in Fig. 2(b). Lastly, the D_{be} term is the bending energy of the estimated thin plate spline transformation at the end of iteration. In [3], the three types of distance measures are combined with a leave-one-out learning procedure. However, within each type, distance measures from all samples are simply summed, yielding an overall distance measure used by the nearest neighbor classifier:

$$D_{all} = 1.6 D_{ac} + D_{sc} + 0.3 D_{be}$$

3. Classification based on a fixed shape

The nearest neighbor classifier effectively weighs the information on each sample point equally. Yet, usually some parts of the shape are better in telling apart two classes. For instance, in the case of distinguishing between 3s and 8s, the left side of the shape is more distinctive than the right side of the shape. Putting more weight on the left side of the shape will presumably give a better distance measure.

At first sight, this suggests a supervised learning approach to weigh the components of the distance measure based on labeled examples. However, an important distinction sets us apart from a global approach to learn the distance measure: our distance measure between shape A and

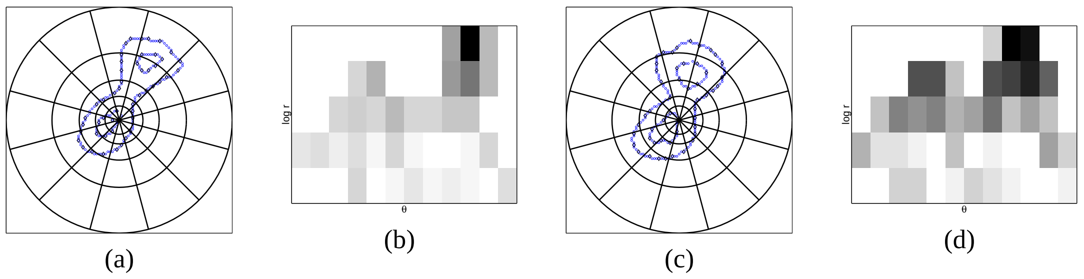

Figure 1. (a) On the contour of the digit eight, the shape contexts are computed with respect to the circled sample points. (b) the log-polar histogram that has 5 bins for the polar direction and 12 bins for the angular direction. Each bin contains a count of the edge points falling into that bin. (c) shape context of a corresponding point on another digit. (d) the histogram is similar to (b),the corresponding point on the other shape.

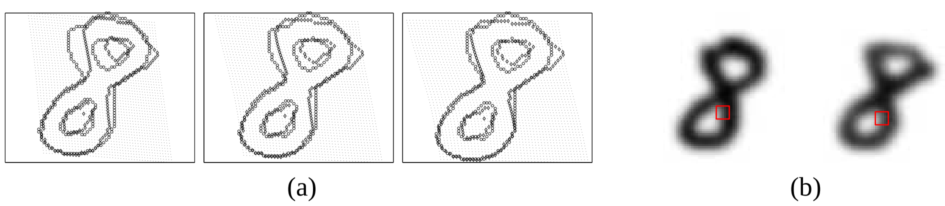

Figure 2. (a)Three iterations in the shape context – thin plate spline (SC+TPS)iteration (b) The shapes are aligned by thin plate spline warping. In an image patch around corresponding points, we then compute the sum of squared difference in intensity D_{ac}.

B is a vector that contains D_{sc}, D_{ac} and D_{be} terms. Each entry in the vector reflects the closeness of the most look-alike points on shape B based on a given sample point on shape A. The ordering of sample points on shape A is arbitrary and no clear correspondence can be made for shapes are very different, for instance between the shape of a 0 and a 1. This makes it impossible to attach a global meaning to the i-th entry of the distance vector.

What we do instead is to anchor on a fixed prototype shape. With this restriction we can learn a proper weighting. We compare all other shapes against the prototype using SC+TPS iteration. This attaches a “distance vector” to all shapes (with self-distance being zero). A typical situation is to have 100 sample points, which results in 100 D_{sc} terms, 100 D_{ac} terms and one term of D_{be}. At this point, we want to weigh the components of this 201 dimensional vector appropriately so as to maximize discriminant power. A number of methods are available for this purpose. Examples include discriminative adaptive nearest neighbor, described in [8]; hierarchical mixture of experts, described in [10]; and binary or multiway logistic regression. Since the training corpus consists of high-dimensional feature vector and many examples (5000-15000), we opt for methods that are light in computation. Trials on small dataset show that binary logistic regression is both accurate and fast.

On example of applying binary logistic regression is as follows: given an anchor shape, we divide the training set into two parts: those with the same label as the anchor shape (A) and the rest (B). We then attach the posterior of 1 to the distance vectors in A and a posterior of 0 to those in B. Given the distance vectors and the posterior, the logistic regression fits a weighting on the distance vector that maximize the likelihood of the attached posterior:

$$P(Y = 1|x) = \frac{1}{1 + \exp(-w^T x - b)}$$

where w is the weight vector and b is an intercept (baseline).

This is so-called “one-vs-all” procedure. However, as noted by [15] and other multiway classification practices, it is often hard for a classifier to distinguish class i from the rest of the classes. Proper separation may require complex, non-linear boundary. An easier task is to distinguish between class i and class j, resulting in what’s called a “pair-wise” classifier [7]. To this end, we alter the training procedure: divide the training set into $N - 1$ parts $B_1, B_2, ..., B_{N-1}$, each of a distinct class other than that of the prototype. We then train a logistic regressor for each of B_i, with a posterior of 1 on distance vectors in A and a posterior of 0 on those in B_i, and ignore the rest of the training set. For a prototype A, this yields nine classifiers of the type "A vs. class i". In contrast, one-vs-all method would just give one classifier of the type "A vs. the rest". In both cases, we call those prototype-based classifiers as "local experts". In next section, we address the question of how to combine the outputs from the local experts.

4. Combining outputs from binary classifiers

4.1. Error correcting output codes

When a query digit comes along, each local expert claims whether the query digit belongs to its class or not (by a "soft" answer in the form of posterior), the simplest way is to take the expert with the maximum response and assign its label to the query digit. This resembles nearest neighbor, with a distorted distance measure. Nearest neighbor is known to perform well with a large number of prototypes (asymptotically, NN performs no worse than twice the minimum Bayes risk [5]). However, with a limited number of prototypes, because NN ignores all the responses but the closest expert, it is sensitive to occasional error from individual local experts.

A better way to combine the responses from the local experts, suggested by recent literature [6] and others, is to treat the response vector as a corrupted codeword that has originated from canonical responses of the true class type. The task is then to decode the response vector so as to recover the class type that the query comes from. This is the theme of "error-correcting output codes" and it is applicable to both variants of local expert: one-vs-all and all-pairs.

[6] studied the performance of different choices of the "coding scheme". In particular, the one-vs-all classifier, viewed as a coding scheme, produces a Hamming distance of 2 between ideal prototypes of two classes. The Hamming distance is defined as the number of places where a pair of 0-1 vectors differ. For the one-vs-all classifier, a class i prototype produces an ideal response vector of $(0, .., 0, 1, 0, .., 0)$ where 1 is placed at the ith entry. Therefore, for $i \neq j$, a class i and a class j response vector differ in two places. In the case of all-pairs classifier, the ideal response vector is a ternary vector of $\{-1, 0, 1\}$ where a 1 denotes positive response, a -1 denotes negative response, and a 0 denotes "don't care". For a pair of ideal response vectors from two difference classes, the generalized Hamming distance is now $4n - 4$ where n is the number of classes. This widened discrepancy between response vectors of different classes makes the overall classifier less prone to errors of an individual classifier, and more capable of telling apart queries of different classes. As we shall see in section 5, this is verified by the experimental results.

[6] also studies different decoding methods, which are ways to combining the outputs from base classifiers to obtain the label. We choose to implement the "L^1 loss-based" decoding because (1) experimental results show that it performs on par with more sophisticated, exponential loss based methods; (2) it has the intuitive interpretation of being the average of positive responses minus the average of negative responses.

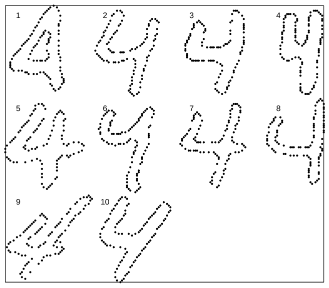

Figure 3. Prototypes selected for a digit class using the K-medoid method on rough distance measures

4.2. Multiple prototypes for each class

The problem of digit classification is characteristic of high intra-class variation. There is much variety in people's way to write some digit, say a 4 or a 7. This necessitates the use of multiple prototypes for each digit class to capture all the variations. In this work, we use the method of K-medoid [8] to select representative prototypes from each class. Within each digit class, we compute pairwise distance measure by simply summing over the D_{sc} terms (this gives us a rough estimate of how far two shapes are apart). We then use this distance measure in the K-medoid algorithm, with K being 1 up to 10. This method has captured most of the variations occurred within a class. An example of selected prototypes is shown in Fig. 3. Given multiple prototypes for a class, we adapt the implementation of ECOC that trains one classifier for each combination of *classes*. Instead, we train one classifier based on each *prototype*: for example, in the case of all-pair method, we train with respect to a prototype of class i *all* the examples of class i and a different class j. This is different than treating each prototype as a distinct sub-class and apply straightforward all-pair method to the induced problem of $K \cdot N$ classes (where K is the number of prototypes for each class and N is the number of classes). The latter method requires dividing the dataset into sub-categories, which can be difficult and is an unnecessary step if all we want is the broad class label.

The outputs from classifiers based on each prototype are then combined using the "L^1 loss-based decoding", by taking the average of positive responses and subtract from it the average of the negative responses. The class that has the highest score is reported as the label of the query. For the

	0	1	2	3	4	5	6	7	8	9
0	478	1	0	1	0	0	2	0	1	0
1	0	533	6	0	1	0	0	1	0	0
2	2	2	450	4	0	0	0	4	4	0
3	1	1	2	493	0	3	0	4	4	0
4	0	4	0	0	476	0	1	0	2	8
5	2	0	0	5	1	448	2	0	2	3
6	0	1	0	0	1	1	473	0	1	0
7	0	1	6	0	4	0	0	499	1	5
8	5	1	0	5	0	2	0	1	467	2
9	1	1	0	4	3	0	1	2	2	458

error rate $= 2.55\%$

Table 1. Error rate and confusion matrix from a sample run. The rows are the true label and the columns are the reported labels. This is all-pairs classifier trained on 15000 examples for 5 prototypes per digit. Error rate is based on 5000 test cases.

one-vs-all method, since there's no negative response, only the average of the postive responses is considered.

5. Results on digit dataset

5.1. Accuracy

We use the MNIST dataset [13] for test. This dataset has been tried on many pattern recognition algorithms such as neural network, nearest neighbors, support vector machines, etc. A comparison of algorithms can be found in [21]. Some of the best performing algorithms include pre-processing of the digit such as deskewing and jittering. Others train on deformed version of the digits so as to achieve a certain degree of invariance under transformation. In our work, the use of the shape context distances takes care of these variability and allows us to concentrate on the study of learning techniques.

The error rate of a given algorithm is computed as the number of mislabeled test digits divided by the total number. The shape context method using 3-NN and 20000 prototypes [3] has reported the lowest test error below 1%.

More important to the application of prototype-based classifiers is its performance on small number of prototypes. The method in [3] has reported error rate of 2.5% on a few hundred prototypes. A typical run of our method, illustrated in table 1, shows that we can achieve the same level of accuracy with about 10 times fewer prototypes.

5.2. Speed and scalability

Efficiency is important for practical use of the method. The cost of time of our system is divided into the training stage and the online stage. The training stage involves prototype selection using K-medoid, computing distance vectors using SC+TPS iteration, and training logistic classifiers (using iterative reweighted least squares method). The online stage consists of two steps: (1) matching with the prototypes using SC+TPS and (2) evaluate the logistic classifiers based on the outputs from (1). The combined cost of the online stage is much lower than that of the training stage since only a few number of prototypes need to be matched to. On Pentium IV 2GHz machine with sufficient memory, the training on 15000 examples takes about four hours, and the testing on 7000 examples takes less than an hour. By construction of the algorithm, the cost of time is linear in the number of prototypes, and linear in the number of training examples.

The accuracy also scales with added number of prototypes, as the error rate drops from 7% to 3% when the number of prototypes per digit class grows from 1 to 10 (Fig. 4, all-pairs method). The error rate can be driven down even further by adding more prototypes. Eventually, we expect the system to perform on par with previous nearest-neighbor based approach that obtained the lower error rate.

A similar situation occurs with added number of training examples (Fig. 5). In this case, the system saturates as enough training examples have been supplied. Fig. 5 shows this is the case for the equal-weighting classifier, which has far fewer parameters (3 weights on the features) than the other two (201 weights). In contrast, the other two methods can overfit on insufficient training data, e.g. the case of the all-pair method trained on 1K examples in Fig. 5. From experiments, we observe that 10K examples are sufficient for proper training of the system.

5.3. Comparison with simpler methods

Fig. 4 and 5 also plot the performance curve for the "equal-weighting" and "one-vs-all" methods. Equal-weighting refers to the method that sums the distance measures of each type, reducing a 201-dimensional vector to 3-dimensions, and then train the rest of the system using the 3 components of the distance. This puts equal weighting on all of the sample point and is the base we started from. As we expected, the performance is not as good, for the lack of distinction among the sample points.

The "one-vs-all" method is also being compared with. As we mentioned before, the all-pairs approach enhances the contrast between response vectors of different classes and this translates into performance gain for the "all-pair" classifier, shown in Fig. 4 and Fig. 5.

6. Summary and Conclusions

Past work on shape matching and object recognition has shown the strength for either approach. In this work, we attempt to bring together the two: apply advanced learning technique to features obtained by shape context methods. Difficulty arises due to the lack of a global definition of feature vector. Our solution is to learn a local discriminative

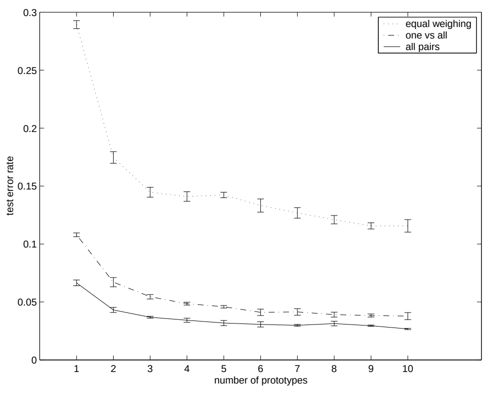

Figure 4. Results on growing number of prototypes. Each run is trained on 10000 examples and tested on 7000 examples, for the three types of classifiers. Test error is estimated by 5-fold cross-validation.

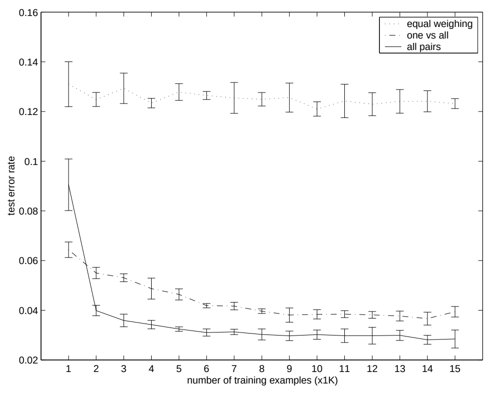

Figure 5. Results on growing number of training examples. Each run is trained on 8 prototypes and tested on 7000 examples, for the three types of classifiers. Test error is estimated by 5-fold cross-validation. In the case of 1K training examples, insufficient training data causes overfitting of the all-pair method.

classifier based on a prototype and obtain a pool of "local experts". We then combine the outputs from local experts using ECOC methods. Results have shown that our method is accurate and efficient, and compares favorable with previous approaches.

References

[1] E. L. Allwein, R. E. Schapire, and S. Yoram. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning*, 1:113–141, December 2000.

[2] Y. Amit, D. Geman, and K. Wilder. Joint induction of shape features and tree classifiers. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(11):1300–1305, November 1997.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(24):509–522, April 2002.

[4] F. L. Bookstein. Principal warps: thin-plate splines and decomposition of deformations. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989.

[5] T. Cover and P. Hart. Nearest neighbor pattern classification. In *IEEE Trans. Inform. Theory*, pages 21–27, 1967.

[6] T. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, pages 263–286, January 1995.

[7] T. Hastie and R. Tibshirani. Classification by pairwise coupling. In *Advances in Neural Information Processing Systems 9: Proceedings of the 1996 Conference*, 1996.

[8] T. Hastie, R. Tibshirani, and J. Friedman. *Elements of Statistical Learning*. Springer Verlag, 2001.

[9] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(9):850–863, Sept. 1993.

[10] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. Technical Report AIM-1440, 1993.

[11] Y. Lamdan, J. Schwartz, and H. Wolfson. Affine invariant model-based object recognition. *IEEE Trans. Robotics and Automation*, 6:578–589, 1990.

[12] L. J. Latecki, R. Lakämper, and U. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, pages 424–429, 2000.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[14] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, pages 193–199, Puerto Rico, June 1997.

[15] V. Roth and K. Tsuda. Pairwise coupling for machine recognition of handprinted japanese characters, 2001.

[16] H. Schneiderman and T. Kanade. A statistical method for 3 dimensional object detection applied to faces and cars. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2000.

[17] D. Sharvit, J. Chan, H. Tek, and B. Kimia. Symmetry-based indexing of image databases. *J. Visual Communication and Image Representation*, 9(4):366–380, December 1998.

[18] M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–96, 1991.

[19] R. C. Veltkamp and M. Hagedoorn. State of the art in shape matching. Technical Report UU-CS-1999-27, Utrecht, 1999.

[20] P. Viola and M. Jones. Robust real-time object detection. In *Second international workshop on statistical and computational theories of vision*, 2001.

[21] L. Y. The MNIST handwritten digit database `http://yann.lecun.com/exdb/mnist/`.