Feature extraction and selection strategies for automated target recognition

W. Nicholas Greene[a], Yuhan Zhang[b], Thomas T. Lu[*c], Tien-Hsin Chao[c]

[a]Princeton University, Princeton, NJ, USA;
[b]California Polytechnic University, Pomona, CA, USA;
[c]Jet Propulsion Laboratory/California Institute of Technology, Pasadena, CA, USA

ABSTRACT

Several feature extraction and selection methods for an existing automatic target recognition (ATR) system using JPLs Grayscale Optical Correlator (GOC) and Optimal Trade-Off Maximum Average Correlation Height (OT-MACH) filter were tested using MATLAB. The ATR system is composed of three stages: a cursory region-of-interest (ROI) search using the GOC and OT-MACH filter, a feature extraction and selection stage, and a final classification stage. Feature extraction and selection concerns transforming potential target data into more useful forms as well as selecting important subsets of that data which may aide in detection and classification. The strategies tested were built around two popular extraction methods: Principal Component Analysis (PCA) and Independent Component Analysis (ICA). Performance was measured based on the classification accuracy and free-response receiver operating characteristic (FROC) output of a support vector machine(SVM) and a neural net (NN) classifier.

Keywords: feature extraction, feature selection, PCA, ICA, GOC, OT-MACH, pattern recognition, computer vision

1. INTRODUCTION

Computer vision, or the science and technology that allows artificial systems to "see" and "understand" their environments, has evolved into a unique field of study as research efforts have attempted to match or exceed the performance of biological solutions, such as the human visual pathway. Despite many advances, artificial solutions are still inferior to their biological counterparts, which leaves much room for improvement. Object detection and recognition, a subset of the field, concerns accurately locating and labeling particular objects or features (edges, cars, etc.) in images and video despite cluttered or "noisy" backgrounds. Automated target recognition (ATR) systems attempt to systematically locate and label targets of interest from sensor networks. This capability can then be applied to everything from smarter search engines to autonomous aircraft and spacecraft. While progress has been made in improving accuracy and speed, ATR performance still lags substantially behind human ability and, therefore, further research is required.

2. BACKGROUND

2.1 JPL's Multi-stage ATR System

A general, two-stage strategy for implementing ATR systems has been developed at JPL that utilizes a grayscale optical correlator (GOC) and a single Optimum Trade-off Maximum Average Correlation Height (OT-MACH) filter in conjunction with adaptive NNs.[1,2] The motivation for this two-stage system stems from the trade-off between speed and accuracy. A fast, but uncritical, analysis of an image can yield false-positives, while a more exhaustive analysis may be computationally slow. The model developed at JPL compromises by first performing a cursory search of the input image using the GOC and OT-MACH filter and then verifying the produced regions-of-interest (ROIs), which may contain false-positives, using an adaptive neural network (NN). This allows the exhaustive analysis (i.e. the NN) to be used sparingly.

Send correspondence to Thomas T. Lu:
E-mail: Thomas.T.Lu@jpl.nasa.gov, Telephone: (818) 345-9513, Fax: (818) 393-4272

During a recognition task, filter training images are first fed into the system so that the parameters for the single OT-MACH filter, which describes targets in the frequency domain, can be determined. The filter is then correlated with the Fourier transforms of the testing images using the GOC and ROIs (where targets may be located) are determined by the Maximum Average Correlation Height for each testing image. The GOC allows for extremely fast computation of the Fourier transform of input images by optical lenses. In addition, OT-MACH filters are well suited to optical processing and balance several conflicting performance measures such as Average Correlation Height (ACH), Average Similarity Measure (ASM), Average Correlation Energy (ACE), and Output Noise Variance (ONV).

As the output ROIs from this first stage may contain false-positives, they undergo further processing and are then fed into an adaptive NN (trained on another set of training images) that verifies whether the ROI is indeed a target or not. Artificial NNs are biologically inspired computational models that offer massive parallelism, learning ability, generalization ability, adaptability, fault tolerance and low energy consumption and thus make them ideal for pattern recognition and object recognition tasks.[3]

This strategy has been improved through adaptive step gradient descent optimization of OT-MACH filters[4] and previously tested with wavelet transforms and feed-forward back-propagation NNs.[5]

2.2 Feature Extraction and Selection

The primary ROI search using the GOC and OT-MACH filter outputs small, 24×48-pixel sections of the original image. These image patches must then be verified as indeed true targets or false-positives by a secondary classifier, such as an NN. Since the ROIs are 24×48 pixels, they are represented in MATLAB as 1,152-dimensional column vectors, each element of which is called a "feature value." These high-dimensional ROI vectors are typically too unwieldy to feed into an NN as is, which necessitates a form of dimension reduction. One approach, called feature selection, reduces the dimension of the ROI vectors by simply selecting a certain important or informative subset of the original features (i.e. rows of the ROI vectors). Feature extraction, on the other hand, maps the original vectors onto a lower-dimensional subspace (i.e. creates new features) where the new, transformed vectors are potentially more useful to the classifier. (e.g. data classes may be more clearly separable).

2.3 Overview of current technology

The goal of feature extraction is to project the original data into a space such that data with different characteristics are grouped away from each other. And hopefully, a simple hyper-surface can be drawn in between two classes with different characteristic as a classifier. Principal component analysis (PCA) and independent component analysis (ICA) are both unsupervised learning tasks that have attracted considerable attention during the past decade for this kind of projection. Many variants of techniques are developed as an extension based on these two strategies. Recent development is mainly focused on how to find clusters with an arbitrary shape distribution (instead of just Normal Gaussian distribution) and fully automate the process of setting clustering parameters.[6]

3. OBJECTIVES AND ORGANIZATION

Starting with the ATR system developed at JPL, this paper tests different methods for improving the verification stage through feature extraction and selection strategies built around PCA and ICA. The training data that would otherwise be used to directly train the final NN is fed into these algorithms, producing a linear transformation that is later applied to the testing data before classification. Descriptions of each strategy will be followed by test results using data sets of various difficulties, a discussion of the results, and possible future work.

3.1 Notation

Throughout this paper, the training ROIs from the first detection stage of the ATR system will be represented as Z-scores in $m \times n$ training data matrix $\mathbf{X}$, where $m = \#$ of features and $n = \#$ of ROIs. Therefore the columns of $\mathbf{X}$ represent the training ROI vectors and the rows of $\mathbf{X}$ have zero mean, unit variance, and represent the features. The testing data matrix $\mathbf{X}_{Test}$ is similarly organized, except it is standardized by the row means and variances of $\mathbf{X}$.

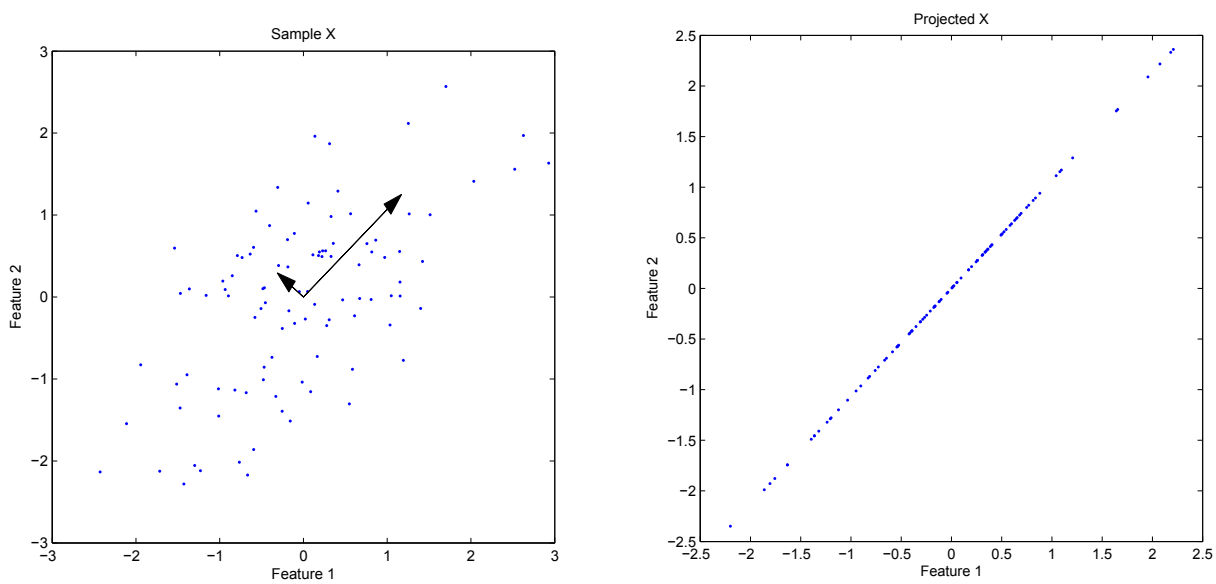
4. FEATURE EXTRACTION AND SELECTION METHODS

4.1 PCA

PCA is a form of dimension reduction developed by Karl Pearson[7] that attempts to preserve the components with the greatest variance of the original data with fewer dimensions. Intuitively, the data is projected onto a basis where each axis represents a direction in which the original data varies greatly. The axes are ordered, with the first axis capturing the greatest amount of variance (i.e. is the direction in which the data varies most), the second axis capturing the second greatest amount of variance variance among those directions perpendicular to the first axis, and so on (See Fig. 1). These axes are called principal components. Feature vectors can then be expressed in this new coordinate system as linear combinations of the principal components (i.e. the basis vectors). Each feature is transformed as a list of weights. Dimension reduction can be achieved by removing the principal components with small variances and preserving the principal components with large variances. By using fewer principal components than the original dimension of the data, the transformed data is an optimal representation in fewer dimensions of the original data in a least square sense.

Given training data $\mathbf{X}$, the transformation algorithm begins by computing the covariance matrix $\mathbf{\Sigma} = \frac{1}{n-1}\mathbf{X}\mathbf{X}^T$ of the features. Here the features (i.e. the rows of $\mathbf{X}$) are considered random variables (it is essential that the they have zero mean). $\mathbf{\Sigma}$ therefore represents the degree to which the features vary with one another.

Using the eigendecomposition $\mathbf{\Sigma} = \mathbf{V}\mathbf{D}\mathbf{V}^T$, the directions in which $\mathbf{X}$ varies can be obtained. The columns of $\mathbf{V}$ represent the eigenvectors of $\mathbf{\Sigma}$ with corresponding eigenvalues in the diagonal matrix $\mathbf{D}$. These vectors are the directions in which $\mathbf{X}$ varies with their corresponding eigenvalues equivalent to the variance displayed in that direction.$\mathbf{V}$ and $\mathbf{D}$ are then arranged in descending eigenvalue order so that the first column of $\mathbf{V}$ represents the direction with the greatest variance, the second column represents the direction with the second greatest



(a) $\mathbf{X}$ with Principal Axes

(b) $\mathbf{V}_k\mathbf{Y} = \mathbf{V}_k\mathbf{V}_k^T\mathbf{X}$

Figure 1: PCA projects data onto a reduced subspace that captures the most significant variance of the data. In (a), a sample $\mathbf{X}$ with two features is plotted along with its principal components (scaled by the variance they capture). In (b) the data is projected onto the first component and then back into two dimensions. (This shows the meaning of the projection more clearly. Note however that in the actual algorithm $\mathbf{Y}$ is used instead of $\mathbf{V}_k\mathbf{Y}$ - i.e. here $\mathbf{k}$ is 1, reducing $\mathbf{Y}$ to 1-dimensional.)

variance, and so on. These are the principal components. $\mathbf{V}$ is then truncated to contain only the principal components with the k greatest variances $\mathbf{V}_k$. $\mathbf{X}$ is then projected onto the subspace spanned by the columns of $\mathbf{V}_k$,

$$\mathbf{Y} = \mathbf{V}_k^T \mathbf{X} \tag{1}$$

The columns of $\mathbf{Y}$ representing the ROI vectors are now of dimension $k < m$. $\mathbf{V}_k$ will later be used to transform the testing data $\mathbf{X}_{Test}$.[8]

4.2 "Whitened" PCA

A simple, but effective, improvement to traditional PCA is an extra "whitening" step that decorrelates the transformed features and normalizes their variances (transforms the data closer to white noise - hence the name. See Fig. 2). Given matrices $\mathbf{D}_k$ and $\mathbf{V}_k$, a new matrix

$$\mathbf{W} = \mathbf{D}_k^{-\frac{1}{2}} \mathbf{V}_k^T \tag{2}$$

is computed. The rows of $\mathbf{W}$ are the principal axes (i.e. columns of $\mathbf{V}_k$) divided by the square root of their corresponding variances (i.e. their standard deviations). This effectively "stretches" the transformed data

$$\mathbf{Y}_W = \mathbf{W}\mathbf{X} \tag{3}$$

so that it varies equally in each principal direction (i.e. $Cov(\mathbf{Y}_W) = \mathbf{I}$). $\mathbf{Y}_W$ no longer approximates $\mathbf{X}$, but rather attempts to "extract" the information contained in $\mathbf{X}$, ultimately the boundaries between targets and false-positives. Similarly, $\mathbf{W}$ will later be used to transform the testing data $\mathbf{X}_{Test}$.

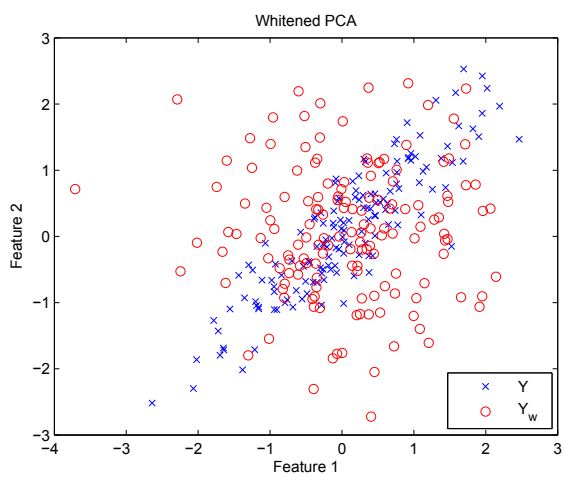
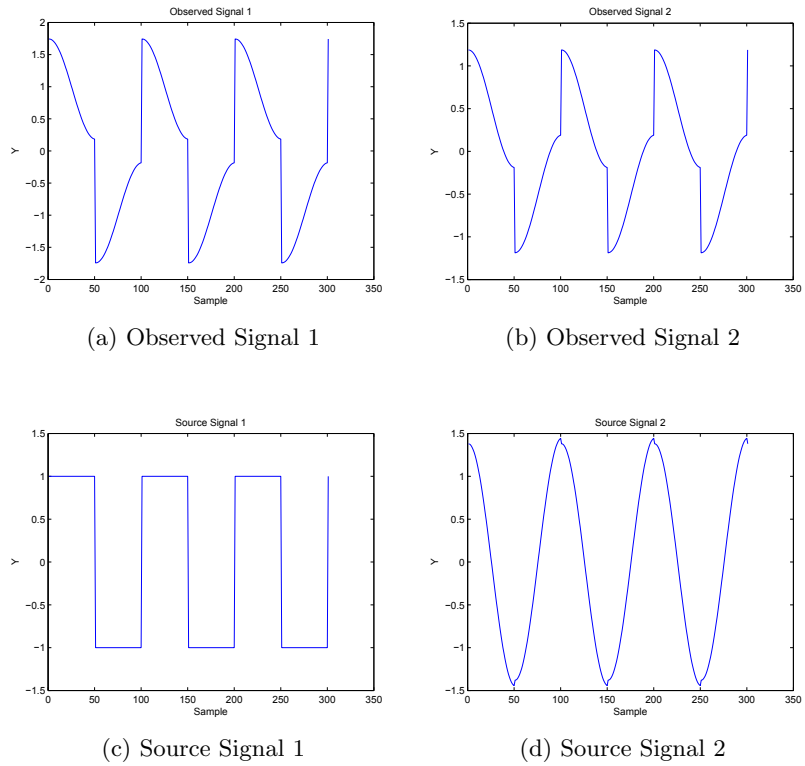


Figure 2: Result of a traditional PCA transform (crosses - highly correlated) and a whitened PCA transform (circles - uncorrelated).

4.3 ICA

ICA is a special case of blind source separation that factors an observed, mixed signal into statistically independent components.[9] For example, consider the "cocktail party problem." Suppose that microphones are placed around a room during a party. The observed signal from each microphone will therefore be a mixture of all the voices in the room. Assuming that there are at least as many microphones as people in the room, ICA allows the observed signals to be separated into the source signals of each voice by assuming that the voices are statistically independent of one another and unmixing the observed signals accordingly (See Fig. 3).



(a) Observed Signal 1 (b) Observed Signal 2

(c) Source Signal 1 (d) Source Signal 2

Figure 3: Given two mixed signals in (a) and (b), ICA is able to extract the source signals (c) and (d). (a) and (b) were generated by linearly combining a square wave and a cosine wave with random weights.

We were motivated to apply ICA to our feature extraction problem by the prospect of representing our ROI vectors in a reduced subspace where each basis vector was statistically indepdent of all the other basis vectors. This would imply that our feature space would be composed of statistically independent features. Assuming that the hypothetical, ideal features representing true-positives and false-positives presented to the ATR system would be independent of one another, projecting our testing data matrix $\mathbf{X}_{Test}$ onto an ICA produced basis may allow for easier target detection and classification.

Given training data $\mathbf{X}$, we assume that its features are mixed observations of statistically independent source features. In other words,

$$\mathbf{X} = \mathbf{AS} \tag{4}$$

which implies

$$\mathbf{S} = \mathbf{A}^{-1}\mathbf{X} = \mathbf{WX} \tag{5}$$

Each column of $\mathbf{X}$ (an ROI vector) can be unmixed by $\mathbf{W}$ to form the corresponding column of $\mathbf{S}$ (an ROI vector with statistically independent features).

We utilized the FastICA algorithm developed by Hyvärinen,[10] which computes $\mathbf{w}_i$ (i.e. the i-th row of $\mathbf{W}$) iteratively by maximizing the non-Gaussianity of the projections $\mathbf{w}_i^T\mathbf{X}$. The central limit theorem poses that large sums of independent random variables (i.e. the rows of $\mathbf{S}$) will approximate Gaussian distributions (i.e. the rows of $\mathbf{X}$). Therefore, a signal's non-Gaussianity compared to another signal is a good metric for its statistical independence. FastICA measures non-Gaussianity using a negentropy approximation

$$J(Y) = H(Y_{Gauss}) - H(Y) \approx (\mathbb{E}[G(Y)] - \mathbb{E}[G(Z)])^2 \tag{6}$$

where Y is a random variable, Y_{Gauss} is a Gaussian random variable with the same mean and variance as Y, and Z is a Gaussian random variable with zero mean and unit variance. G is a chosen contrast function. We used

$$G(u) = \frac{1}{a}\log(\cosh(au)) \tag{7}$$

with the default setting of $a = 1$.

Note, however, that this algorithm does not reduce the number of features. It merely transforms the observed features into statistically independent ones. Whitened PCA is used by FastICA as a preprocessing step if dimension reduction is desired.

5. METHODOLOGY AND RESULTS

5.1 Data

We tested the strategies using two different types of data: sonar images of underwater objects and digital images taken from a boat at sea(See Fig. 4). Sonar images may contain multiple targets and usually have a very noisy background. Underwater objects can come in different sizes, shapes, and textures, making detection and classification tough. The boat images usually consist a scene of water, sky, boats and even buildings on shore. The images are captured under various degrees of illumination, under different weather conditions, with or without close-to-shore objects, and may be tilted due to the physical movement of the boat that houses the camera. Each image usually contains at most one or two boats as the true targets to be tracked. A boat as a target in one data set could be considered as a non-target in another data set - it depends on the tester to define what boat to be tracked.

(a) Boat Sample 1 (b) Boat Sample 2 (c) Sonar Sample

Figure 4: Examples of the data used.(a) contains a ship at sea under the sunshine. The sparkles on the water may lead to false positives. (b) contains a boat captured far away from the camera. The target may shrink in size in the image due to distance. (c) contains two underwater objects on the ocean floor captured via sonar.

5.2 Classifiers

Classification was performed using two different types of classifiers: support vector machines (SVMs) and NNs. These classifiers operate differently and have their own strengths and weaknesses. SVMs find separating hyperplanes between training data classes that maximize the margin (distance) between the classes. They are inherently linear classifiers, but can be used for nonlinear classification by projecting the training data to a higher-dimensional space before the separating hyperplane is found.[11] We employed a radial-basis-function (RBF) kernel for nonlinear classification. NNs are biologically inspired computational models that find nonlinear boundaries between data classes by iterative learning paradigms. While NNs have been used for JPL's ATR system before with good results, the randomness introduced during the NN training phase may obscure the effects of the different feature extraction strategies. SVMs, in turn, are not subject to a training phase built on random factors and can thus isolate the differences between extraction strategies. However, there is no guarantee that SVM performance is greater than NN performance, so the extraction strategies were tested using both classifiers. Performance was measured by comparing average true-positive-rates and false-positive rates (scaled to show average detection performance versus the average number of false-positives per test image) over seven different data sets (see Table 1 and 2).

Table 1: Data used for SVM tests. Training and testing ROIs are denoted for each class (e.g. Class 0/Class 1/Class 2). Class 0 is the non-target class in all tests.

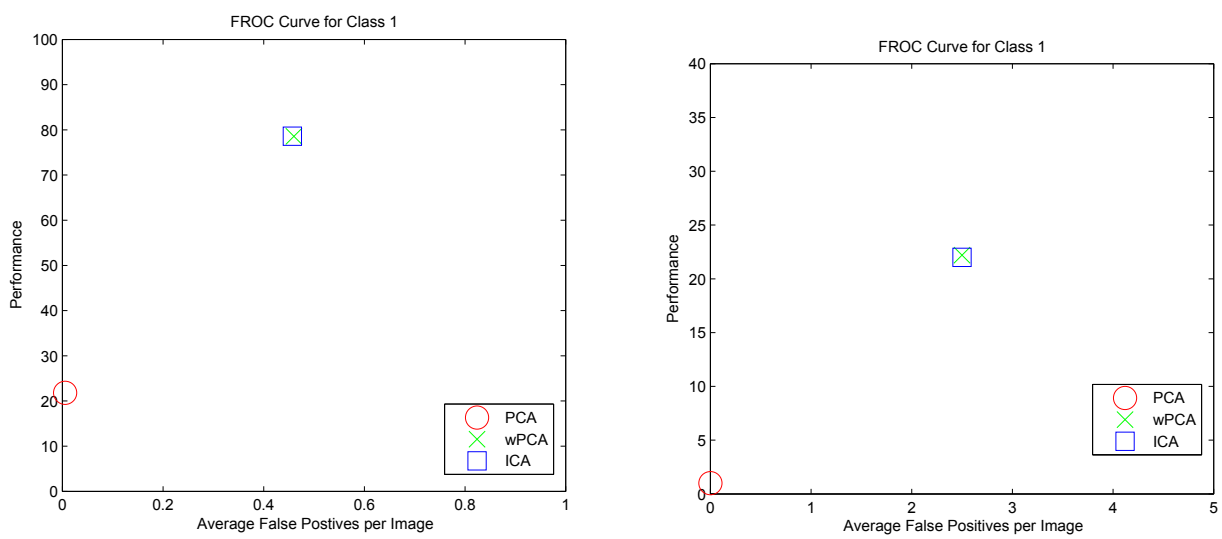
Name	Difficulty	# Trials	# Features	# Training ROIs	# Testing ROIs
June 5	Easy	5	12	200/40	83155/445
June 9	Medium	20	12	80/20/20	118302/68/59
Sonar 1	Hard	20	12	40/20	17160/105

Table 2: Data used for NN tests. Training and testing ROIs are denoted for each class (e.g. Class 0/Class 1/Class 2). Class 0 is the non-target class in all tests.

Name	Difficulty	# Trials	# Features	# Training ROIs	# Testing ROIs
Nov 5 Thu 11 28 54	Easy	50	18	9/21	56/137
Nov 5 Thu 09 26 46	Easy	50	18	281/31	26677/239
Oct 22	Medium	50	18	434/42	15769/85
Sonar new	Hard	50	18	260/42	5881/217

5.3 SVM Results

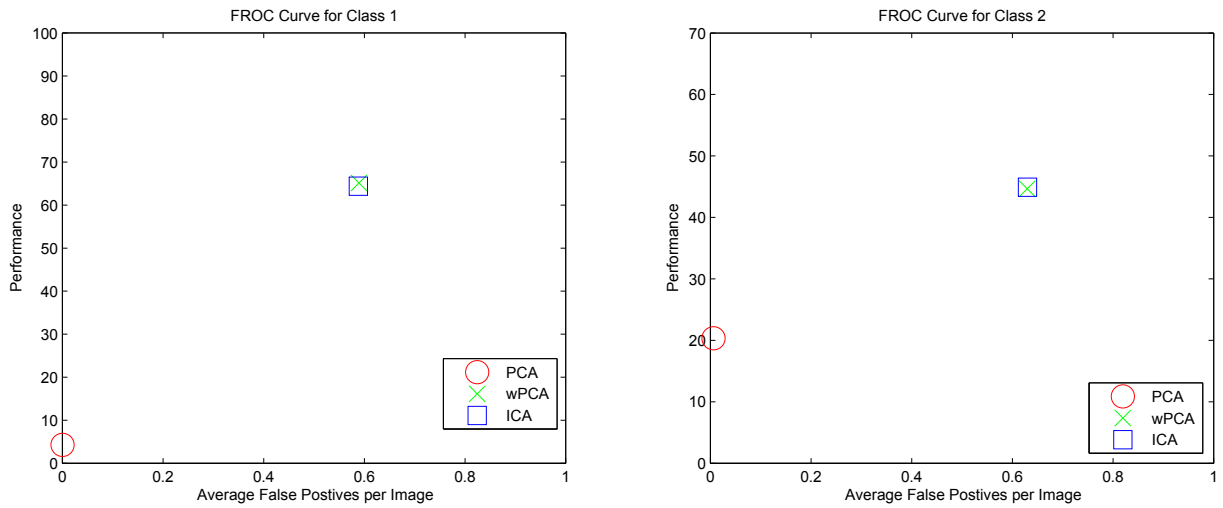
The SVM tests were performed on each dataset by first manually selecting the ground truths to be detected and applying an OT-MACH filter to identify the ROIs. A random subset of the ROIs are then used as training data. They are transformed via the different feature extraction strategies and then used to train RBF SVMs. The rest of the ROIs are then classified with these SVMs and the true-positive-rate and the false-positive-rate are recorded. This is performed for a certain number of trials after which the mean true-positive-rates and false-positive-rates are calculated. Results can be seen in Fig. 5 and 6.



(a) June 5

(b) Sonar 1

Figure 5: SVM results for the June 5 and Sonar 1 data sets. Ideal performance would entail 100% performance and 0 average false-positives per image. Notice that whitened PCA and ICA perform nearly identically.



(a) June 9 - Class 1

(b) June 9 - Class 2

Figure 6: SVM results for the June 9 data set. Performance is gathered for each target class versus all other classes. Ideal performance would entail 100% performance and 0 average false-positives per image. Notice that whitened PCA and ICA perform nearly identically.

5.4 NN Results

The NN tests were performed on each dataset by first applying an OT-MACH filter to the images to identify ROIs. Then, a training set of around 15-30% of the ROIs is manually selected with the rest of the ROIs used as testing data. The training set is then transformed via the different feature extraction strategies and then used to train a NN. Since NN training using the Levenberg-Marquardt algorithm[12,13] depends on random factors, multiple NNs are trained (50 in all) with the same training data and the best is picked by comparing FROC curves. FROC curves juxtapose the true-positive rate versus the false-positive rate (scaled to show detection performance versus the average number of false-positives per test image) over the sensitivity range of the NN. The performance of each NN is measured by the area underneath the FROC curve and the NN with the best performance for each extraction strategy is presented in Fig. 7 and 8.

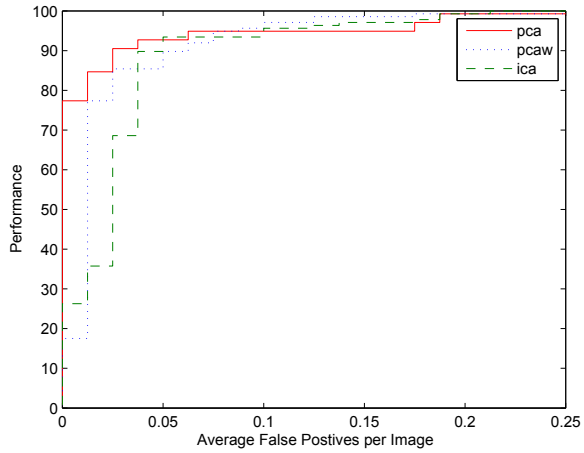
6. DISCUSSION AND FUTURE WORK

Our results showed that no one strategy was a decisive winner on these datasets. However, some insights into the relative strengths and weaknesses of these algorithms can be gleaned.

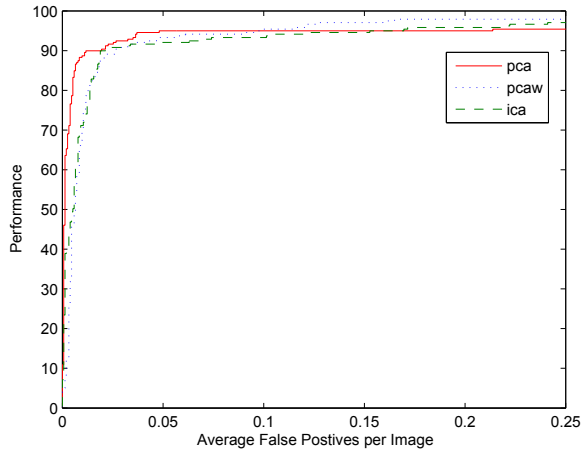
6.1 PCA

PCA is based on the assumption that the class separation may be more significant in the direction with longer range of differences. Notice that rather than separating classes, this method attempts to approximate an accurate reconstruction of an ROI in a lower dimension. In our case, each axis is an "eigen image" of an ROI and a linear combination of these eigen images gives us back an approximation of the original ROI. Usually the axes with the most significant variances are the lower frequency components, where it describes the background. So, the dimension reduction with the top axes may not be relevant in separating the classes if the target does not take up the entire ROI space. Training based on those features could result classification based on the background intensity instead of the object itself, which makes it fails to generalize. The poor results of PCA using the SVM could be due to this overfitting problem - i.e. the hyperplane found by the SVM to separate the training data does not necessarily generalize beyond the training set.

In addition, the separating boundary between the data classes in the PCA space may not be linear. While we used an RBF kernel for the SVM, which allows to it to find nonlinear, spherical decision boundaries, the

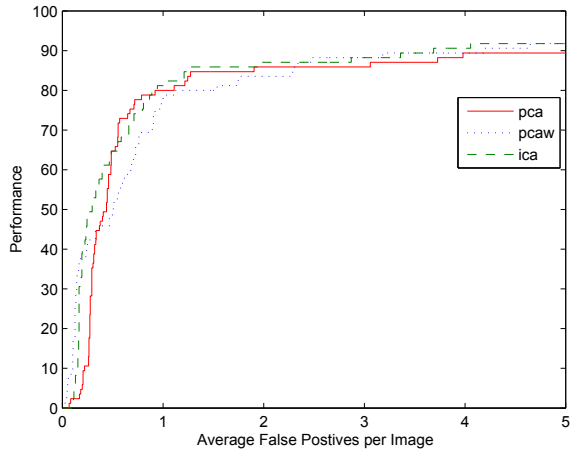


(a) Nov 5 Thu 11 28 54, between 0 to 0.25 average false-positives per image

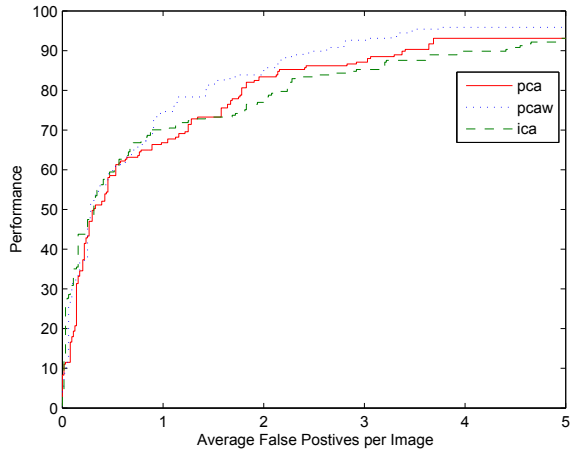


(b) Nov 5 Thu 09 26 46, between 0 to 0.25 average false-positives per image

Figure 7: a) The FROC curve for data set "Nov 5 Thu 11 28 54" (easy): all three feature extraction methods could reach 100% true target detection within less than 1 false positive per image, with ICA having the leading performance. b) The FROC curve for data set "Nov 5 Thu 09 26 46" (easy): all methods can recognize around 98% of true targets with just 1 false positive per image misclassified. In this case Whitened PCA works a little better. In general, for the boat images without many variations (easy), all 3 methods will work around the same performance.



(a) Oct 22, between 0 to 5 average false-positives per image



(b) Sonar new, between 0 to 5 average false-positives per image

Figure 8: a) The FROC curve for data set "Oct 22" (medium): ICA has a better performance within the range of having 90% true target detected, although Whitened PCA certainly performs better when more than 10 false positives per images are allowed. b) The FROC curve for data set "Sonar new" (hard): Whitened PCA may have an advantage over the other two methods.

RBF kernel may be too general for our specific data. For example, the best decision boundary in the PCA space may be an ellipsoid, while the RBF SVM specializes in spherical boundaries. Smarter nonlinear classifiers such as NNs appear to do much better.

So, principle components with the most significant variance, combined with RBF SVMs, may not be ideal for nonlinear class seperation.[14]

6.2 PCA vs. Whitened PCA

From the SVM results, whitening provides a substantial performance boost to traditional PCA, with gains of up to 60% with only a small increase in false-positives. This is most likely due to the uneven distribution of the number of non-target ROIs to target ROIs. With around 99% of the data in all the data sets belonging to the 0 (non-target) class, simply approximating the data structure in a lower dimensional space does not aid in distinguishing the target ROIs. In all three SVM tests, PCA showed a negligible false-positive rate, suggesting that the SVM simply classified all ROI vectors as class 0 (this was later confirmed). The decorrelation step, which transforms the ROI data into a hypersphere in the whitened PCA space, is absolutely crucial in correctly identifying any significant number of targets. This spherical data cluster in the whitened PCA space may be easier for the RBF SVM, which specializes in spherical decision boundaries, to classify the targets.

6.3 ICA vs. Whitened PCA

As can be seen in all three SVM test results, ICA and whitened PCA performed nearly identically. This was quite puzzling at first, but may be explained by the intricacies of FastICA. As mentioned before, ICA as used in this sytem does not reduce the dimension of the data - it simply represents the data with statistically independent basis vectors (i.e. it expresses the data in a new coordinate system, not a new subspace). While statistically independent features is desirable, the fact that whitened PCA is used as a preprocessing step in FastICA effectively "locks" the subspace where the data will be projected. The basis vectors computed for the PCA transformed space and ICA transformed space are indeed different, but they may span the exact same subspace, meaning that the end structure of both sets of transformed data is identical (although may be scaled or rotated with respect to the coordinate system), leading to similar performance. This means that whitened PCA and ICA should perform exactly identically. While we observed extremely small differences in performance (equivalent to misclassifying one or two ROIs), we believed that these were most likely due to the fact that each scheme performs PCA with different MATLAB functions. For example our whitened PCA scheme uses the MATLAB function "princomp" while the ICA scheme utilizes the functions "cov" and "eig" to perform its PCA calculation. This difference can lead to slightly different (though nearly identical) principal components. More tests using were conducted with each algorithm using the same MATLAB functions and identical results were produced, confirming our intuitions.

The implications of this in relation to NN classifiers is discussed in the next section.

6.4 SVMs vs. NNs

The direct comparison of classifiers was not an original objective of this paper, which is why different datasets were used for each classifier. The feature extraction methods used here should be independent to the classifier, which is employed as a tool to measure the class separations in data (thus the effectiveness of the feature extraction methods). However, the differences in performance between the SVM tests and the NN tests may be important for future work.

First, the difference in traditional PCA performance when using an SVM and when using a NN classifier is dramatic. This may simply be due to the inherent differences in the classifiers. The PCA transformed data may not offer optimal nonlinear class separation, as the SVM results show. However, the nonlinear and randomized nature of the NN may allow it to find an adequate separation boundary, even without an apparent class separation (i.e. the NN may be "smarter" than the SVM).

This intuition may also factor in to the comparison of whitened PCA and ICA. The SVM tests suggest that whitened PCA and ICA should perform identically, since data in the whitened PCA space and in the ICA space should be scaled and rotated versions of each other (Figs. 5 and 6). However, the NN tests show substantial variations in performance (Figs. 7 and 8). This could be attributed to the inherent randomness of the NN training stage and to our method of picking optimal NNs. We train 50 NNs for each extraction scheme and pick the best one to compare extraction methods. This means that the NN picked for whitened PCA and the NN picked for ICA can be different, beyond the scaling and rotation difference between the whitened PCA and ICA

spaces. We suspect that if the same NN separating boundary was somehow scaled and rotated appropriately, it could be used to classify whitened PCA data and ICA data and produce identical FROC curves*. If this were the case, then the observed variation between whitened PCA and ICA using NNs could be considered negligible, which in turn suggests from our results that traditional PCA performs just as well as the other schemes using a NN. Furthermore this would also explain why traditional PCA performs so poorly using an SVM: the feature extraction methods used here do not significantly affect NN performance.

6.5 Future Work

The simplicity and all-around performance of whitened PCA makes this algorithm a good starting block for future feature extraction methods. Combining this strategy with some mutual information metric or additional transformation step in the whitened PCA domain may be promising. ICA may also prove useful if the FastICA algorithm was changed so that the actual ICA algorithm was performed before whitened PCA. This might allow ICA to extract some information from the data before dimension reduction and decorrelation. The affect these feature extraction methods have on SVM and NN performance can also be investigated.

7. ACKNOWLEDGMENTS

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration (NASA), and was under the sponsorship of the Summer Undergraduate Research Fellowship (SURF) program through JPL and Caltech.

REFERENCES

[1] Zhou, H., Hughlett, C., Hanan, J., and Chao, T., "On the development of filter management module for grayscale optical correlator," *SPIE Conference* **5437**, 87–94 (2004).

[2] Lu, T., Hughlett, C., Zhou, H., Chao, T., and Hanan, J., "Neural network post-processing of grayscale optical correlator," *SPIE Conference* **5908** (2005).

[3] Jain, A., Mao, J., and Mohiuddin, K., "Artificial neural networks: A tutorial," *IEEE Computer* **29**, 31–44 (1996).

[4] Johnson, O., Edens, W., Lu, T., and Chao, T., "Optimization of ot-mach filter generation for target recognition," *SPIE Conference* **7340** (2009).

[5] Ye, D., Edens, W., Lu, T., and Chao, T., "Neural network target identification system for false alarm reduction," *SPIE Conference* **7340** (2009).

[6] Bohm, C., Faloutsos, C., and Plant, C., "Outlier-robust clustering using independent components," *SIGMOD international Conference* , 185–198 (2008).

[7] Pearson, K., "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine* , 559–572 (1901).

[8] Shlens, J., "A tutorial on principla component analysis," *http://www.snl.salk.edu/ shlens/pub/notes/pca.pdf* (2009).

[9] Comon, P., "Independent component analysis, a new concept?," *Signal Processing, Elsevier* **36**, 287–314 (1901).

[10] Hyvarinen, A., "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Transactions on Neural Networks* **10**, 626–634 (1999).

[11] Boser, B. E., Guyon, I. M., and Vapnik, V. N., "A training algorithm for optimal margin classifiers," in [*5th Annual ACM Workshop on COLT*], 144–152, ACM Press (1992).

[12] Levenberg, K., "A method for the solution of certain non-linear problems in least squares," *The Quarterly of Applied Mathematics* **2**, 164–168 (1944).

[13] Levenberg, K., "An algorithm for least-squares estimation of nonlinear parameters," *SIAM Journal on Applied Mathematics* **11**, 431–441 (1963).

[14] Guyon, I. and Elisseeff, A., "An introduction to variable and feature selection," *The Journal of Machine Learning Research* **3**, 1157–1182 (2003).

*Note that the separating hypersurface in a certain subspace found by a NN is specific to that coordinate system and cannot be directly applied to the same data in the same subspace, but in a scaled or rotated coordinate system.