Vessel classification in overhead satellite imagery using learned dictionaries

Katie Rainey, Shibin Parameswaran, Josh Harguess, John Stastny

Katie Rainey, Shibin Parameswaran, Josh Harguess, John Stastny, "Vessel classification in overhead satellite imagery using learned dictionaries," Proc. SPIE 8499, Applications of Digital Image Processing XXXV, 84992F (15 October 2012); doi: 10.1117/12.928875

SPIE.

Event: SPIE Optical Engineering + Applications, 2012, San Diego, California, United States

Vessel classification in overhead satellite imagery using learned dictionaries

Katie Rainey°⋆, Shibin Parameswaran⋆, Josh Harguess⋆ and John Stastny

Space and Naval Warfare Systems Center Pacific, 53560 Hull Street, San Diego, CA 92152-5001

ABSTRACT

Recognition and classification of vessels in maritime imagery is a challenging problem with applications to security and military scenarios. Aspects of this problem are similar to well-studied problems in object recognition, but it is in many ways more complex than a problem such as face recognition. A vessel's appearance can vary significantly from image to image depending on factors such as lighting condition, viewing geometry, and sea state, and there is often wide variation between ships of the same class. This paper explores the efficacy of several object recognition algorithms at classifying ships and other ocean vessels in commercial panchromatic satellite imagery. The recognition algorithms tested include traditional classification methods as well as more recent methods utilizing sparse matrix representations and dictionary learning. The impacts on classification accuracy of various pre-processing steps on vessel imagery are explored, and we discuss how these algorithms are being used in existing systems to detect and classify vessels in satellite imagery.

Keywords: object recognition, ship classification, dictionary learning, sparse representation

1. INTRODUCTION

The success of any object recognition algorithm is heavily dependent on the data set to which the algorithm is applied. Recently developed algorithms have demonstrated near perfect accuracy on some standard face data sets, but do not perform as well on collections of images of more general objects.[1] The purpose of this work is to study the effectiveness of several published methods for object recognition for the task of vessel classification. Recognition of ships and other vessels in ocean imagery is a challenging problem; vessel images can vary significantly in resolution, illumination conditions, viewing angle, and background, and two ships of the same type may possess different features.

Automated vessel classification algorithms are relevant to many military and security applications. The RAPid Image Exploitation Resource (RAPIER®), developed by Space and Naval Warfare (SPAWAR) Systems Center Pacific,[2] provides automated ship detection in overhead satellite imagery as well as in full-motion video from unmanned aerial systems in support of image analysts. RAPIER returns ship detects in the form of small images chipped out of a larger video or satellite image (see Figure 1). Algorithms to recognize the ship class or type of each image chip are necessary to provide increased functionality to RAPIER and greater benefit to the analyst.

The problem of ship classification has been previously studied by the authors.[3,4] In 3, well-studied facial recognition techniques are applied to a small set of ship images that have been pre-processed in order to approximate the cropped and aligned face data sets usually tested with these algorithms. In 4, in an effort to avoid the pre-processing step, the sparse representation-based classification method proposed in 5 is modified to accommodate unprocessed ship imagery, which are of non-uniform size. Both of these lines of inquiry are continued in this work. Various algorithms for object recognition are tested on several data sets of vessel imagery collected from RAPIER detects from overhead electro-optical satellite imagery. These data sets have undergone various stages of pre-processing, and the possible accuracy improvement to be obtained on pre-processed data sets is explored. The work in 4 is extended by applying learned dictionaries to the sparse representation-based method, and the improvement offered by this addition is studied. Several methods are identified which obtain high accuracy rates when applied to ship data that has been manually pre-processed (rotated, cropped and aligned). The application

Author e-mails: {kate.rainey, shibin.parameswaran, joshua.harguess, john.stastny}@navy.mil

° Corresponding author

⋆ Equally contributing authors

Applications of Digital Image Processing XXXV, edited by Andrew G. Tescher, Proc. of SPIE
Vol. 8499, 84992F · © 2012 SPIE · CCC code: 0277-786/12/$18 · doi: 10.1117/12.928875

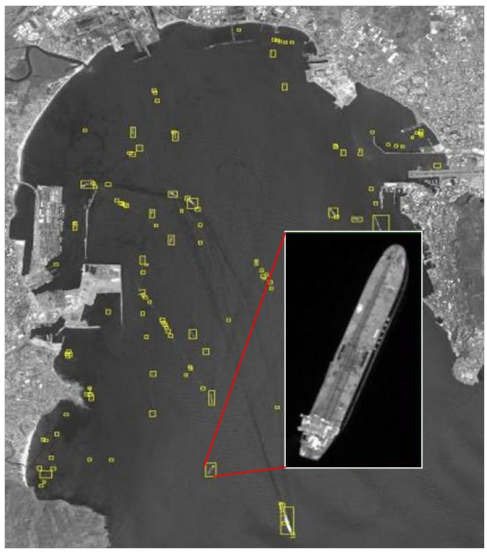

Figure 1. The RAPIER Ship Detection System automatically detects ships and chips them out of larger overhead satellite imagery.

of learned dictionaries is shown to improve the classification results of standard algorithms on unprocessed ship imagery.

This paper is organized as follows. The ship data sets and algorithms tested are detailed further in sections 2 and 3, respectively. Experimental results are stated and discussed in section 4, and the paper is concluded in section 5.

2. VESSEL DATA SETS

Four data sets of overhead ship imagery have been constructed on which to test various classification algorithms. These data sets consist of small grey-scale ship images chipped out of larger electro-optical satellite images by the RAPIER Ship Detection System. The chips have been sorted by hand into appropriate ship type categories and pre-processed to varying degrees. Two hundred images have been collected from each of the following classes: barges, container ships, cargo ships, and tanker ships*. Examples of the classes in this data set, referred to as BCCT200, can be seen in Figure 2.

2.1 Pre-processing

Data sets commonly used to test face recognition algorithms, such as the cropped version of the Extended Yale Face Database,[6] contain images which have been cropped and aligned so that the images are of a uniform size and common features (eyes, etc.) appear in the same location across the set.

Satellite images, however, display ships under various orientations and resolutions. Such variations make it impossible to apply many common feature extraction algorithms (see section 3.1) to the unprocessed ship imagery in BCCT200. Many algorithms of this type which have been successful in the problem of face recognition have also recently been shown to be useful in ship classification.[3] In order to further study the performance of such algorithms for the problem of ship classification, additional data sets have been constructed by applying pre-processing steps to the images in BCCT200.

Points from four regions of each ship image are obtained in order to rotate, crop and resize the original BCCT200 data set. These regions are the bow (front-most part of the ship), stern (rear-most part of the ship), port (left side of the ship) and starboard (right side of the ship). Both the bow and stern points form a line which bisects

*These are general classes of large commercial vessels, chosen by their appearance, and only loosely tied to classifications used in commercial shipping. For example, a container ship is a type of cargo ship, but its appearance is distinctive enough that it warrants its own class for this data set.

(a) (b) (c) (d)

Figure 2. Example images from each class of the BCCT200 data set after resizing: (a) barge, (b) cargo ship, (c) container ship, and (d) tanker.

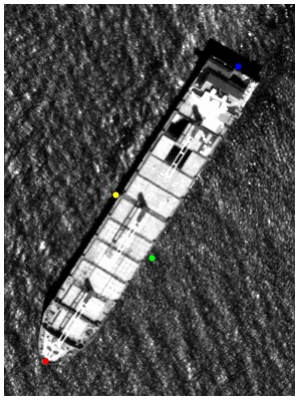

Figure 3. Example ship image with bow (red), stern (blue), port (green) and starboard (yellow) points marked.

the ship into mostly equal halves. The port and starboard points are selected anywhere on the border of their respective sides of the ship. Figure 3 displays an example image of a ship with each of these points marked. For this work the necessary points are obtained manually for each ship, but future research into automating the selection of these points is planned.

Three variations of the BCCT200 data set have been constructed:

- BCCT200-rot - images are rotated
- BCCT200-crop - images are rotated and cropped
- BCCT200-resize - images are rotated, cropped, aligned and resized.

The three pre-processing steps are described in detail below.

2.2 Rotation

The first pre-processing step is to rotate all of the images to a common orientation, arbitrarily selected to be vertical with the bow of the ship pointing up. To rotate the ships, the bow and stern locations are used to calculate the angle difference of the current orientation of the ship versus the desired orientation. Once the angle difference is known, MATLAB® is used to rotate the image to the desired orientation. An example image after this rotation is performed is shown in Figure 4(b).

2.2.1 Cropping

The bow, stern, port and starboard locations are used to automatically determine the cropped region of the image. The region is centered so that the bow and stern points are in the middle of the image, and a percentage of the pixels surrounding the ship are removed: 20% from the sides and 10% from the top and bottom. After

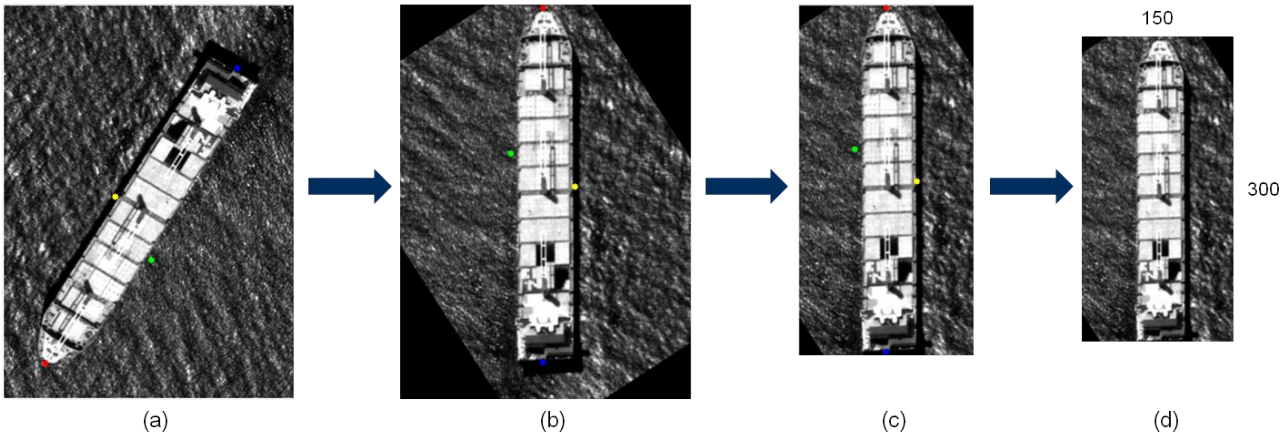

Figure 4. Example ship images from each data set: (a) unprocessed; (b) rotated; (c) rotated and cropped; and (d) rotated, cropped, aligned and resized.

this step, not all ship images will appear the same since some bow and stern locations may be close to the edges of the image. An example image resulting from the rotation and cropping is shown in Figure 4(c).

It is not expected that simply rotating the ships to a common orientation will improve the recognition accuracy, especially if rotation invariant features are used for feature extraction. However, cropping the image may remove some noise from the background and thus improve recognition results. Furthermore, it is an essential step towards forming the aligned ship data set.

2.2.2 Aligning and resizing

The final step in forming the aligned and resized data set is to align the locations of the bow and stern, and resize the images to common dimensions. This is accomplished as follows. By using the bow, stern, port and starboard locations of each ship image, the maximum ratio of height to width of the ship data set is obtained. For BCCT200, that ratio is 2:1, so a common height of 300 pixels and a common width of 150 pixels are selected. The bow and stern locations are used to determine the appropriate rescaling needed for each of the images and then the image is either zero padded or cropped to a size of 150 pixels wide. An example image resulting from rotation, cropping and resizing is show in Figure 4(d).

3. RECOGNITION ALGORITHMS

Most systems in computer vision for object classification consist of two distinct components: feature extraction, in which an image is transformed into a representative set of features, and classification, in which an image is assigned to one of several given categories. Several feature extraction methods coupled with various classification algorithms have been tested on ship data to study their effectiveness for the vessel classification task and to understand the challenges of this problem in greater clarity. This section lists and explains the feature extraction methods and classification algorithms used in this study.

3.1 Feature extraction

Feature extraction is a necessary and critical step in any machine learning problem. This step serves to reduce the dimensionality of the problem, as well as to extract relevant or discriminative features that will impact the success of the stages that follow, namely detection or classification. For this study, a number of such algorithms commonly used in object and face recognition have been chosen and tested for their effectiveness in vessel recognition.

3.1.1 Subspace projection methods

Many feature extraction methods work by linearly projecting data into a lower dimensional subspace. Principal component analysis (PCA) is used widely in face and object detection and recognition applications.[7] PCA finds an orthogonal transformation of the data such that the individual features in the resulting space are linearly uncorrelated and the dimensions of new coordinate system are ordered with respect to the amount of variance captured by the respective projections. This provides an easy criteria for feature selection and thus for dimension reduction.

Multilinear PCA is an extension of PCA which uses tensors, or multilinear arrays, to represent the projection from image to a lower dimensional feature space.[8] The idea is to determine a mulitlinear projection for the image to preserve some of the 2D structure of the image.

Linear discriminant analysis (LDA) aims to find a linear transformation that increases class separability in the transformed space.[9] LDA achieves this by finding a lower dimensional space where the inter-class scatter is maximum and intra-class scatter is minimum.

Independent component analysis (ICA) is a generalization of PCA which attempts to identify high-order statistical relationships between pixels to form basis vectors for discrimination between classes. For these experiments, the pixels are treated as random variables and the images as outcomes, as described in 10.

Features may be extracted randomly, instead of by a carefully selected transformation. Wright *et al.*[5] project data to a lower dimensional linear subspace by means of a random transformation matrix with entries independently sampled from a zero-mean normal distribution and each row normalized to unit length. This computationally simple method has been shown[5] to be as effective in some applications as more complex methods such as PCA.

3.1.2 Local features

Other feature extraction methods use histograms of local features to represent an image. Local binary patterns (LBP) is a method which has been used by several researchers[11–14] to capture features useful in face recognition. LBP operates on gray-scale texture to characterize the local spatial structure of the image texture. A 2-bit pattern code is computed by comparing a central pixel with its neighbors:

$$LBP = \sum s(p_n - p_c)2^N, \tag{1}$$

where p_c is the gray value of the central pixel, p_n is the value of its neighbors, s is evaluated to be 0 if $p_c > p_n$ and 1 otherwise and N is the total number of neighbors involved in the computation. Another parameter controls the radius of the neighborhood. After computing the LBP of each pixel in the image, a histogram is built to represent the whole image. Then, a comparison of sample and model histograms is done, typically using a nonparametric statistical test of goodness-of-fit. A variant of LBP called hierarchical multiscale LBP (HMLBP) is employed in this study. HMLBP has been shown in 11 to outperform other LBP methods for face recognition.

Dalal and Triggs[15] first introduced histograms of oriented gradients (HOG) to detect humans in static images. Since the introduction of this feature extraction method, HOG has been used in numerous computer vision applications, such as face recognition.[16–18] HOG captures the distribution of local intensity gradients to represent the image. First, the image gradients are found. Next, a chosen grid size is used to divide the image into cells and histograms of the gradients in each cell are computed using majority voting. Finally, the feature vector of the image is formed by concatenating the histograms from each of the cells.

Bag of (visual) words (BOW) is a feature extraction approach inspired by the bag of words representation used in text classification tasks.[19] In text applications, BOW treats a document as a collection of words independent of each other, ignoring the order and context in which the words are used. In image applications, BOW represents an image as a histograms of its local features, using feature descriptors such as Scale-Invariant Feature Transform (SIFT).[20] For this work, SIFT descriptors are extracted from a set of training imagery and clustered to form the vocabulary of visual "words". A histogram is formed by binning the SIFT descriptors based on their proximity to the visual vocabulary words. Unlike the methods listed above, a BOW representation can be used for tasks involving image samples with non-uniform sizes as their input. For this reason, this feature has been used extensively in object recognition tasks.

3.2 Classification methods

After feature extraction, the next step in the object recognition process is classification. The role of a classification algorithm is to assign a class label to a given image. For this work, three types of classification methodologies have been chosen – nearest neighbor, support vector machines and sparse representation-based classification.

3.2.1 Nearest neighbor

Nearest neighbor (NN) is one of the oldest, but simplest, classification methods used in computer vision.[21] A similarity or distance measure s is calculated between a feature vector representing a test image and each of the feature vectors from a set of training images. The test image is assigned to the class to which belongs the closest or most similar training image (its "nearest neighbor"). That is, if y is a feature vector extracted from a test image and $\mathcal{T}$ is the set of feature vectors of training images,

$$class(y) = class(x_j), \text{ where } x_j = \arg\min_{z \in \mathcal{T}} s(y, z).$$

The similarity measurement may be a true metric-based distance in the mathematical sense, such as the familiar Euclidean distance (L_2 norm),

$$s_{Euc}(x, y) = \|x - y\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2},$$

or the Manhattan distance (L_1 norm),

$$s_{Manh}(x, y) = \|x - y\|_1 = \sum_{i=1}^{n} |x_i - y_i|.$$

Another metric-based distance is the Mahalanobis distance, which captures the correlations between features within the measurement. This distance is defined as

$$s_{Mahal}(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)}, \tag{2}$$

where S is the covariance matrix of x, which is calculated from the feature vectors of the training images. When the covariance matrix is equal to the identity matrix, this distance is equivalent to the Euclidean distance.

Similarity measurements not based on metrics may also be employed, such as the cosine similarity that calculates the cosine of the angle between two vectors,

$$s_{Cos}(x, y) = 1 - \cos(\theta) = 1 - \frac{x \cdot y}{\|x\| \|y\|}.$$

Histogram distances If the feature extraction method used returns a histogram, such as BOW, a measure of the difference between histograms (which can be considered as probability distributions) may be used.

The chi-squared distance, which comes from statistical distribution tests, is given by

$$s_{ChiSq}(p, q) = \frac{1}{2} \sum_{i=1}^{n} \frac{(p(i) - q(i))^2}{p(i) + q(i)}$$

where p and q are histograms normalized so that the sum of their components is equal to 1.

The Kullback-Leibler distance, or KL divergence, between two probability distributions (or normalized histograms) p and q is defined as

$$D_{KL}(p, q) = \sum_{i=1}^{n} p(i) log \frac{p(i)}{q(i)}.$$

This measure is taken from the area of information theory and has been shown to be useful in many applications for measuring the difference between two distributions. For classification purposes, a symmetric version of this distance is often considered. This work uses a symmetric variation of this distance,

$$s_{KL}(p,q) = D_{KL}(p,q) + D_{KL}(q,p).$$

The earth mover's distance (EMD)[22] is informally defined as the minimum amount of work needed to change one histogram (or probability distribution) into the other. Essentially, the more "dirt" that has to be moved to make one histogram match another, the larger the value of the EMD. A more detailed and formal definition is given in 22.

Histogram based distances are usually only applicable to feature vectors which may be expressed as histograms or probability distributions. In particular, in this work the histogram distances described here are only applied to the histogram-based feature extraction methods described in section 3.1.2.

3.2.2 Support vector machine

A support vector machine (SVM) is a type of linear classifier that is designed to maximize the margin of the decision boundary between positive and negative examples, or support vectors.[23] This method has been used extensively and successfully in many application areas, including face and object recognition. In some cases, the linear classification boundaries learned by SVMs may not be adequate to reliably classify data from real-world applications. This is overcome by using what is known as a kernel trick, allowing SVMs to learn highly non-linear decision boundaries by projecting the input data to a higher dimensional feature space. In this work, both a linear kernel and a non-linear radial basis function (RBF) kernel are considered.

3.2.3 Sparse representation-based classification

Sparse representation-based classification (SRC), first described for face recognition,[5] attempts to classify an image by representing it as a sparse linear combination of basis vectors compiled from a set of training images. Features† extracted from each of k classes are compiled to form a dictionary

$$\mathbf{A} = [\mathbf{A}_1 \cdots \mathbf{A}_k] = [\mathbf{v}_{1,1} \cdots \mathbf{v}_{1,N_1} \cdots \mathbf{v}_{k,1} \cdots \mathbf{v}_{k,N_k}], \tag{3}$$

where the columns of $\mathbf{A}_i$ represent the images from the ith class. This method assumes that if a test image vector $\mathbf{y}$ belongs to class i, and if the dictionary $\mathbf{A}$ is sufficiently overcomplete, then $\mathbf{y}$ can be written as a linear combination of the training samples in $\mathbf{A}_i$:

$$\mathbf{y} = \alpha_{i,1}\mathbf{v}_{i,1} + \cdots + \alpha_{i,N_i}\mathbf{v}_{i,N_i}$$

for some scalar weights $\alpha_{i,j} \in \mathbb{R}$. Then $\mathbf{y}$ may be represented as a sparse linear combination of the training images in $\mathbf{A}$, that is, $\mathbf{y} = \mathbf{A}\mathbf{x_0}$, where

$$\mathbf{x_0} = [0, \cdots, 0, \alpha_{i,1}, \cdots \alpha_{i,N_i}, 0, \cdots, 0].$$

Most of the entries of x_0 are zero since there is no contribution from any class except for the ith class, hence it is sparse.

Given a dictionary $\mathbf{A}$ and a test vector $\mathbf{y}$, the task is then to solve

$$\mathbf{x_0} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{such that} \quad \mathbf{A}\mathbf{x} = \mathbf{y}, \tag{4}$$

where $\|\mathbf{x}\|_0$ is the number of non-zero entries in a vector $\mathbf{x}$. The problem of finding a solution to (4) is NP-hard, but can be approximated using greedy methods such as orthogonal matching pursuit (OMP).[24] Recent results have shown that if the solution $\mathbf{x_0}$ is sufficiently sparse, the solution to (4) can be found by instead solving

$$\mathbf{x_0} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{such that} \quad \mathbf{A}\mathbf{x} = \mathbf{y}, \tag{5}$$

†The authors of 5 use randomly selected features, aiming to reduce the dimensionality of the problem rather than to locate especially relevant features.

where $\|\cdot\|_1$ is the usual L_1-norm.[25–27] Problem (5) can be solved in polynomial time by standard linear programming methods.[28,29]

In an ideal setting, most of the non-zero elements of $\mathbf{x_0}$ should come from one class, and this class determines the classification of the test image $\mathbf{y}$. However in practice, since many of the coefficients in $\mathbf{x_0}$ are close to, but not exactly, zero, classification is carried out based on the class that minimizes the reconstruction error. For each class i, $\mathbf{A}\delta_i(\mathbf{x_0})$ is calculated, where δ_i is a characteristic function which selects only the elements of a vector associated with the ith class. That is, the entries $\delta_i(\mathbf{x_0})$ are zero except for the coefficients corresponding to class i. If the test image $\mathbf{y}$ belongs to class i, the product $\mathbf{A}\delta_i(\mathbf{x_0})$ should be a good approximation of $\mathbf{y}$. The test image then belongs to the class for which this product returns the minimum L_2 error,

$$class(\mathbf{y}) = \arg\min_i \|\mathbf{y} - \mathbf{A}\delta_i(\mathbf{x_0})\|_2,$$

where $\mathbf{x_0}$ solves equation (4) or (5). Other error measurements may be used instead to determine the best approximation, but the L_2 error has been chosen for these experiments due to its simplicity and relative accuracy.

Dictionary learning The success achieved in 5 with SRC depends heavily on the assumption that the data used to build the dictionary $\mathbf{A}$ are "extensive enough to span the conditions that might occur in the test set." SRC has been shown to be effective on highly pre-processed data sets constructed under controlled conditions and containing relatively few variations within each subject, such as the Extended Yale Database B.[30] Building the dictionary by concatonating all training images, as in equation (3), may quickly lead to prohibitively large dictionaries. A large dictionary is not only less efficient but also disadvantageous for sparse coding. One way to circumvent this problem is to choose a subset of the training samples that best represent each class. Recently research has shown that learning dictionaries for sparse coding can improve signal reconstruction.[31]

The K-SVD algorithm[31] learns a dictionary for use in the L_0 problem (4) by finding a solution to the following optimization problem:

$$\min_{D,X} \sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \leq \lambda, \ \forall i, \tag{6}$$

where the $\mathbf{y}_i$ are input signals, $\mathbf{D}$ is the dictionary, $\mathbf{X}$ is the matrix of sparse codes $\mathbf{x}_i$ and λ is the sparsity parameter. This setting explicitly imposes a strict sparsity constraint while minimizing the total reconstruction error. An L_1 cost function may be used instead to learn dictionaries for problem (5), such as the following proposed in 32:

$$\min_{D,X} \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}_i\|_1 \quad \text{subject to} \quad \sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2 \leq \lambda. \tag{7}$$

4. EXPERIMENTAL RESULTS AND DISCUSSION

For a thorough evaluation of the pre-processing techniques and dictionary learning described above, different features and classification methodologies have been tested. The three classification techniques discussed in section 3.2 have been implemented with the various feature extraction methods detailed in section 3.1. The methods and implementations used are listed in Table 1. The results reported in this section are averages over five different 80%/20% cuts of data, of which the larger portion is used for training. The data splits are created by dividing the available data into five disjoint subsets. For each run, a different subset is held out for testing and the remaining four are used for training. Cross-validation is performed to select the optimal parameters for each data set and feature extraction method combination.

4.1 Effects of learned dictionaries

With the exception of the BCCT200-resize set, the ship images considered here are of non-uniform size, so the only feature extraction method of section 3.1 which can be applied to this data is BOW. For each image in BCCT200, as well as the three pre-processed data sets, BOW histograms are compiled from a 500-word vocabulary clustered from SIFT features extracted from a training set of ship images. Five classification methods are then trained and tested with these BOW feature vectors:

Category	Method	Abbr.	Implementation
Features	Principal Component Analysis	PCA	self
	Linear Discriminant Analysis	LDA	Cai *et al.*[33]
	Independent Component Analysis	ICA	self
	Random Projection	Rand	self
	Multilinear PCA	MPCA	Lu *et al.*[8]
	Hierarchical Multiscale Local Binary Patterns	HMLBP	Guo *et al.*[11]
	Histogram of Oriented Gradients	HOG	self
	Bag of Words	BOW	VLFeat toolbox[34]
Nearest Neighbor	Euclidean Distance	Euc	Dollár[35]
	Manhattan Distance	Manh	
	Mahalanobis Distance	Mahal	
	Cosine Distance	Cos	
	Chi-Squared Distance	ChiSq	
	Kullback-Leibler	KL	
	Earth Mover's Distance	EMD	
Support Vector Machine	Support Vector Machine	SVM	LIBSVM toolbox[36]
Sparse Coding	Sparse Representation-based Classification	SRC	SPAMS toolbox[32,37]
Dictionary Learning	K-SVD/L_0-Dictionary	—	
	L_1-Dictionary	—	

Table 1. A list of algorithms and implementations used.

- SVM;
- the L_0 formulation of SRC (equation (4))
  - with the dictionary compiled from training data (equation (3)), and
  - with a dictionary learned from equation (6); and
- the L_1 formulation of SRC (equation (5))
  - with the dictionary compiled from training data (equation (3)), and
  - with a dictionary learned from equation (7).

The results of these experiments are shown in Table 2. For the SVM experiments, the "one vs. one" approach is used for solving multi-class problems, and a parameter search is conducted using both linear and RBF kernels to select the best parameters for each feature extraction method.

The results in Table 2 show that pre-processing steps do not result in significantly improved classification accuracy using SVM or SRC. This is as expected, since the SIFT features used to build the BOW histograms should be rotation-invariant, and the feature extraction method used does not require alignment. The most encouraging result is obtained by applying dictionary learning to the L_1 SRC problem on BCCT200-orig. The learned dictionary provides a nearly five percentage point improvement over the same method using a dictionary built from training data. Indeed, in most cases, dictionary learning is able to improve SRC results, though the best overall result for SRC is obtained on the BCCT200-resize data set using a dictionary of training vectors. The dictionary learning methods implemented for this work are off-the-shelf algorithms not tailored to the classification problem. Therefore, results may be improved by learning more discriminating dictionaries, such as those proposed in 1. Though the SRC results, even with learned dictionaries, are comparable with SVM, the

	SVM	SRC-L_0 – Eq. (4)		SRC-L_1 – Eq. (5)	
		Tr. – Eq. (3)	Dict. – Eq. (6)	Tr. – Eq. (3)	Dict. – Eq. (7)
BCCT200-orig	76.3	73.8	74.8	70.9	75.6
BCCT200-rot	74.5	73.9	74.6	74.8	72.6
BCCT200-crop	74.6	73.6	74.4	74.8	74.6
BCCT200-resize	76.8[a]	73.9	74.6	76.1[b]	74.1

Table 2. Results of SVM and SRC classification on BOW features extracted from the original BCCT200 and the pre-processed data sets. The L_0 and L_1 formulations of SRC are used with dictionaries built from training vectors (Tr.) as well as with learned dictionaries (Dict.). The average percentage accuracy over five runs is shown. Superscripts indicate experiments which also appear in Table 3.

	SRC	SVM	Nearest Neighbor						
			Euc	Manh	Mahal	Cos	ChiSq	KL	EMD
BOW	76.1[b]	76.8[a]	65.9	67.1	50.6	65.9	67.0	55.3	45.1
PCA	81.4	77.1	79.8	79.2	76.9	79.8	N/A	N/A	68.6
LDA	62.0	74.1	71.9	72.1	69.3	71.9	N/A	N/A	63.8
ICA	80.5	77.4	78.1	79.1	77.8	78.1	N/A	N/A	65.5
Rand	78.3	73.9	72.0	71.3	48.8	72.0	N/A	N/A	52.9
MPCA	82.9	79.1	77.8	78.6	70.8	77.8	N/A	N/A	63.8
HMLBP	78.5	**90.8**	60.1	60.5	21.9	60.1	66.8	73.1	33.3
HOG	82.4	81.6	75.8	76.8	61.8	75.8	76.6	76.5	60.0

Table 3. Classification results with SRC, SVM, and NN on the BCCT200-resize data set. The average percentage accuracy over five runs is shown. Superscripts indicate experiments which also appear in Table 2.

SVM results are achieved after an extensive parameter search. Dictionary learning is a relatively new field of research and recent advances may offer greater improvement to ship classification accuracy.

4.2 Classification on aligned and resized data

Table 3 shows the results of classification of aligned and resized data using various types of feature vectors. All of the feature extraction methods listed in section 3.1, including BOW with SIFT feature descriptors, are applied to the BCCT200-resize data set, in which all images are a uniform size, and classification experiments are performed using SVM, SRC (the L_1 formulation without dictionary learning), and NN using various distance measures. Note that the ChiSq and KL distance measures are only applicable to feature types which are normalized histograms, that is, BOW, HMLBP, and HOG.

The results in Table 3 show that even higher accuracy can be obtained on pre-processed — rotated, cropped, aligned, and resized — data, on which feature extraction methods alternative to BOW can be used. Even the more classical methods, such as PCA + NN (which performed at 79.8% accuracy), outperform the best result from the unaligned data (BOW+SVM at 76.3%). HMLBP + SVM performs surprisingly well at 90.8%, almost 15 percentage points higher than the highest classification accuracy obtained on BCCT-orig. This result may be attributed to the ability of HMLBP to capture local features efficiently. Its hierarchical nature may be providing more discriminative class information than other local features like HOG.

4.3 Discussion and future work

The experiments in this work show that data pre-processing may enable significantly improved accuracy for the ship classification problem. Feature extraction with HMLPB, as well as similar local feature extraction methods, should be studied further to better understand what makes them so effective on ship data.

The pre-processing steps applied to the BCCT200-resize data set are performed manually for these experiments, but this is not practical in ship classification applications such as the RAPIER Ship Detection System. RAPIER is able to automatically extract certain information about ship detects, including length, width and heading. This additional information may be leveraged to enable the development of automated algorithms to crop, align, and resize ship data prior to classification.

Additional research into BOW feature extraction and dictionary learning for the ship classification problem is planned. The BOW algorithms used for these experiments are built from local SIFT feature descriptors, but other methods for extracting local features may improve classification results. Classification accuracy using SRC on unprocessed data may also be improved by the use of more discriminative learned dictionaries better suited to the classification problem.

5. CONCLUSION

This work has examined the effectiveness of several image classification and feature extraction algorithms on ship imagery. Several methods have been identified which are very effective for classifying pre-processed ship images that have been aligned and resized to uniform dimensions. In an effort to avoid the need to align and resize data, classification methods which can be employed on unprocessed data have also been explored. Though these are not as accurate as the methods used on aligned and resized images, possible improvements have been identified. The results presented in this work demonstrate the baseline effectiveness of well-known classification algorithms applied to the ship classification problem, illuminating the future directions for this problem.

ACKNOWLEDGMENTS

This work is supported by the Office of Naval Research. The authors would like to thank Luke Barrington, Shay Har-Noy, Alexey Castrodad and Edward Bosch for many helpful discussions.

REFERENCES

[1] Jiang, Z., Lin, Z., and Davis, L. S., "Learning a discriminative dictionary for sparse coding via label consistent K-SVD," IEEE Conference on Computer Vision and Pattern Recognition, 1697–1704 (2011).

[2] Buck, H., Sharghi, E., Guilas, C., Stastny, J., Morgart, W., Schalcosky, B., and Pifko, K., "Enhanced ship detection from overhead imagery," Proc. SPIE 6945 (2008).

[3] Harguess, J. and Rainey, K., "Are face recognition methods useful for classifying ships?," Applied Imagery Pattern Recognition Workshop, 1–7 (2011).

[4] Rainey, K. and Stastny, J., "Object recognition in ocean imagery using feature selection and compressive sensing," Applied Imagery Pattern Recognition Workshop, 1–6 (2011).

[5] Wright, J., Yang, A., Ganesh, A., Sastry, S., and Ma, Y., "Robust face recognition via sparse representation," IEEE Trans. Pattern Anal. Mach. Intelligence 31(2), 210–227 (2009).

[6] Lee, K., Ho, J., and Kriegman, D., "Acquiring linear subspaces for face recognition under variable lighting," IEEE Trans. Pattern Anal. Mach. Intelligence 27(5), 684–698 (2005).

[7] Turk, M. and Pentland, A., "Eigenfaces for recognition," Journal of Cognitive Neuroscience (1991).

[8] Lu, H., Plataniotis, K. N., and Venetsanopoulos, A. N., "MPCA: Multilinear principal component analysis of tensor objects," IEEE Trans. on Neural Networks 19(1), 18–39 (2008).

[9] Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J., "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," IEEE Trans. Pattern Anal. Mach. Intelligence 19(7), 711–720 (1997).

[10] Bartlett, M. S., Movellan, J. R., and Sejnowski, T. J., "Face recognition by independent component analysis," IEEE Trans. Neural Netw. 13, 1450–1464 (2002).

[11] Guo, Z., Zhang, L., Zhang, D., and Mou, X., "Hierarchical multiscale LBP for face and palmprint recognition," IEEE International Conference on Image Processing, 4521–4524 (2010).

[12] Ahonen, T., Hadid, A., et al., "Face description with local binary patterns: Application to face recognition," IEEE Trans. Pattern Anal. Mach. Intelligence , 2037–2041 (2006).

[13] Zhang, G., Huang, X., Li, S., Wang, Y., and Wu, X., "Boosting local binary pattern (LBP)-based face recognition," in [Advances in Biometric Person Authentication], Lecture Notes in Computer Science 3338, 179–486, Springer Berlin / Heidelberg (2005).
[14] Liao, S., Zhu, X., Lei, Z., Zhang, L., and Li, S., "Learning multi-scale block local binary patterns for face recognition," Advances in Biometrics , 828–837 (2007).
[15] Dalal, N. and Triggs, B., "Histograms of oriented gradients for human detection," IEEE Conference on Computer Vision and Pattern Recognition 1, 886–893 (2005).
[16] Albiol, A., Monzo, D., Martin, A., Sastre, J., and Albiol, A., "Face recognition using HOG–EBGM," Pattern Recognition Letters 29(10), 1537–1543 (2008).
[17] Hu, Y., Zeng, Z., Yin, L., Wei, X., Zhou, X., and Huang, T., "Multi-view facial expression recognition," 8th IEEE International Conference on Automatic Face & Gesture Recognition, 1–6, IEEE (2008).
[18] Sivic, J., Everingham, M., and Zisserman, A., "'Who are you?' — Learning person specific classifiers from video," IEEE Conference on Computer Vision and Pattern Recognition, 1145–1152, IEEE (2009).
[19] Yang, J., Jiang, Yugang Hauptmann, A. G., and Ngo, C.-W., "Evaluating bag-of-visual-words representations in scene classification," in [Multimedia Information Retrieval], 197–206 (2007).
[20] Lowe, D. G., "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision 60, 91–110 (2004).
[21] Duda, R., Hart, P., and Stork, D., [*Pattern Classification*], vol. 2, John Wiley & Sons, New York (2001).
[22] Rubner, Y., Tomasi, C., and Guibas, L., "The earth mover's distance as a metric for image retrieval," International Journal of Computer Vision 40(2), 99–121 (2000).
[23] Schölkopf, B., Smola, A. J., Williamson, R. C., and Bartlett, P. L., "New support vector algorithms," Neural Comput. 12(5), 1207–1245 (2000).
[24] Tropp, J. A. and Gilbert, A. C., "Signal recovery from random measurements via orthogonal matching pursuit," IEEE Trans. Inform. Theory 53, 4655–4666 (2007).
[25] Donoho, D. L., "For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution," Comm. Pure Appl. Math 59, 797–829 (2004).
[26] Candès, E. J., K., R. J., and Tao, T., "Stable signal recovery from incomplete and inaccurate measurements," Comm. Pure Appl. Math 59(8), 1207–1223 (2006).
[27] Candès, E. and Tao, T., "Near-optimal signal recovery from random projections: Universal encoding strategies?," IEEE Trans. Inf. Theory 52(12), 5406–5425 (2006).
[28] Chen, S. S., Donoho, D. L., and Saunders, M. A., "Atomic decomposition by basis pursuit," SIAM J. Sci. Comput. 20, 33–61 (Dec. 1998).
[29] Donoho, D. L. and Tsaig, Y., "Fast solution of l1-norm minimization problems when the solution may be sparse," (2006).
[30] Lee, K., Ho, J., and Kriegman, D., "Acquiring linear subspaces for face recognition under variable lighting," IEEE Trans. Pattern Anal. Mach. Intelligence 27(5), 684–698 (2005).
[31] Aharon, M., Elad, M., and Bruckstein, A., "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," IEEE Trans. Signal Process. 54, 43110–4322 (Nov. 2006).
[32] Mairal, J., Bach, F., Ponce, J., and Sapiro, G., "Online learning for matrix factorization and sparse coding," J. Mach. Learn. Res. 11, 19–60 (Mar. 2010).
[33] Cai, D., He, X., and Han, J., "SRDA: An efficient algorithm for large-scale discriminant analysis," IEEE Trans. on Knowl. and Data Eng. 20(1), 1–12 (2008).
[34] Vedaldi, A. and Fulkerson, B., "VLFeat: An open and portable library of computer vision algorithms." `http://www.vlfeat.org/` (2008).
[35] Dollár, P., "Piotr's Image and Video Matlab Toolbox (PMT)." `http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html`.
[36] Chang, C.-C. and Lin, C.-J., "LIBSVM: A library for support vector machines," ACM TIST 2, 27:1–27:27 (2011). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
[37] Mairal, J., Bach, F., Ponce, J., and Sapiro, G., "Online dictionary learning for sparse coding," Proceedings of the 26th Annual International Conference on Machine Learning, 689–696, ACM, New York, NY, USA (2009).