NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

VESSEL CLASSIFICATION FROM OVERHEAD IMAGERY USING THE RANDOM FOREST ALGORITHM

by

Thomas J. Shaheen

December 2020

Thesis Advisor: Monique P. Fargues
Second Reader: Roberto Cristi

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** December 2020	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
VESSEL CLASSIFICATION FROM OVERHEAD IMAGERY USING THE RANDOM FOREST ALGORITHM

5. FUNDING NUMBERS

6. AUTHOR(S) Thomas J. Shaheen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Classification of satellite ship imagery is an active topic of research, and multiple types of classifiers have been considered over the years. This study explores the viability of the random forest algorithm in vessel type classification and compares performance to that obtained in earlier work by Rainey et al., published in a 2012 *SPIE Proceedings* article, and by Parameswaran and Rainey, published in a 2015 *SPIE Proceedings* article. Random forest is advantageous due to its relative ease of use, resistance to overfitting, and built-in model validation. Results indicate that random forest performance is comparable to or better than time-tested machine learning methods, such as support vector machines, when applied to preprocessed vessel images. Feature extractors that capture spatial information yielded highest accuracies. Previous work has indicated that the visual bag of words (VBOW) representation is flexible and effective in feature coding the vessel images. Therefore, in this work various weighting schemes augmented the VBOW, which was evaluated on both original and preprocessed vessel image datasets as input to the random forest, with limited success.

14. SUBJECT TERMS random forest, BCCT200, support vector machines, local binary pattern, histogram of oriented gradients, visual bag of words, VBOW, vessels	**15. NUMBER OF PAGES** 73
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

VESSEL CLASSIFICATION FROM OVERHEAD IMAGERY USING THE RANDOM FOREST ALGORITHM

Thomas J. Shaheen
Lieutenant, United States Navy
BS, U.S. Naval Academy, 2014

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 2020

Approved by: Monique P. Fargues
Advisor

Roberto Cristi
Second Reader

Douglas J. Fouts
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Classification of satellite ship imagery is an active topic of research, and multiple types of classifiers have been considered over the years. This study explores the viability of the random forest algorithm in vessel type classification and compares performance to that obtained in earlier work by Rainey et al., published in a 2012 *SPIE Proceedings* article, and by Parameswaran and Rainey, published in a 2015 *SPIE Proceedings* article. Random forest is advantageous due to its relative ease of use, resistance to overfitting, and built-in model validation. Results indicate that random forest performance is comparable to or better than time-tested machine learning methods, such as support vector machines, when applied to preprocessed vessel images. Feature extractors that capture spatial information yielded highest accuracies. Previous work has indicated that the visual bag of words (VBOW) representation is flexible and effective in feature coding the vessel images. Therefore, in this work various weighting schemes augmented the VBOW, which was evaluated on both original and preprocessed vessel image datasets as input to the random forest, with limited success.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

BCCT-200	Barges, Cargo ships, Container ships, Tankers 200
CART	Classification and Regression Tree
CF	collection frequency
CLBP	Completed Local Binary Pattern
CNN	convolutional neural network
DoG	Difference of Gaussian
ECOC	error correcting output code
HMLBP	Hierarchical Multiscale Local Binary Pattern
HOG	histogram of oriented gradients
IDF	inverse document frequency
IG	information gain
LBP	local binary pattern
LOGOR	Log Odds Ratio
MFL	Multiple Features Learning
MI	mutual information
MS-CLBP	Multiscale Completed Local Binary Pattern
OR	Odds Ratio
PCA	Principal Component Analysis
PIDF	probabilistic inverse document frequency
RAPIER	RAPid Image Exploitation Resource
RF	relevance frequency
SIFT	scale invariant feature transform
SPM	Spatial Pyramid Matching
SRC	Sparse Representation-based Classification
SURF	speeded-up robust feature
SVM	support vector machine
TF	term frequency
U-SURF	Upright speeded-up robust feature
VBOW	visual bag of words

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Professor Fargues for her patience and dedication throughout this process. Her guidance has made this study a fruitful learning experience. I would like to thank Professor Cristi for his teaching and insight. Finally, I am eternally grateful to my parents for their unwavering love and support.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The U.S. Navy has an abiding interest in technologies and processes that enhance Maritime Domain Awareness, defined by the International Maritime Organization as "the effective understanding of any activity associated with the maritime domain that could impact upon the security, safety, economy, or environment" [1, p. 1]. One key component of Maritime Domain Awareness is vessel classification and identification. A major source of surveillance is satellite imagery, which is becoming more abundant every day. To efficiently classify vessels captured in satellite images, robust machine learning algorithms are needed. The process for training these machine learning algorithms is shown in Figure 1. Supervised learning algorithms used in image classification are trained on labelled image datasets. Images then undergo a variety of levels of pre-processing, to include rotation, cropping, alignment, resizing, and normalization. Pre-processing may be necessary so that certain feature extractors can be applied [2]. The feature extractors reduce the dimensionality of the dataset and extract useful and discriminative attributes of the images [3], such as corners, blobs, and edges. The feature vectors of the images and corresponding labels then are used to build and test the classifier algorithm.

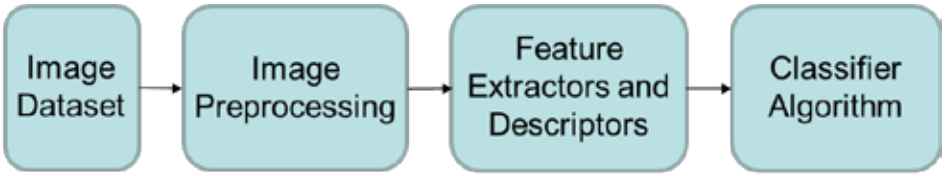


Figure 1. Block diagram of process for training classifiers

One of the challenges of vessel type classification using satellite imagery is that the appearance of vessels in satellite imagery largely depends on lighting conditions, viewing angle, weather and sea state [2]. Additionally, vessels of the same class may exhibit wide variation in features [2].

A potential candidate for vessel classification is the random forest algorithm. It is incredibly versatile, with applications in far-ranging fields including finance and

medicine [4]. More recently, it has become a popular classifier in remote sensing due to its high accuracy and computational efficiency [5]. The random forest has shown success in remote sensing applications such as identifying tree health, mapping of oil spills, and classifying insect defoliation levels [5]. The algorithm requires few hyper-parameters, and can quickly process high dimensional and heterogeneous data. The goal of this research is to explore the efficacy of the random forest algorithm in vessel classification using satellite imagery.

A. DATASET

The data used to train and assess the random forest is BCCT-200, which is composed of grey-scale satellite images of four classes of vessels—barges, cargo ships, container ships, and tankers—consisting of 200 vessel images per type. This dataset was compiled using the RAPid Image Exploitation Resource (RAPIER®), developed by Space and Naval Warfare (SPAWAR) Systems Center Pacific [2]. RAPIER detects ships in satellite imagery and from unmanned aerial system video [2]. This study analyzes two subsets of the BCCT-200:

- BCCT-200_orig: No pre-processing is applied to the original images. These images are various sizes and show vessels in various orientations. They are hereafter referred to as original images.
- BCCT-200_resize: These images have been rotated, cropped, aligned, and resized to 300 x 150 pixels [2]. They are hereafter referred to as resized images.

Examples of original and resized images are shown in Figure 2 and Figure 3, respectively.

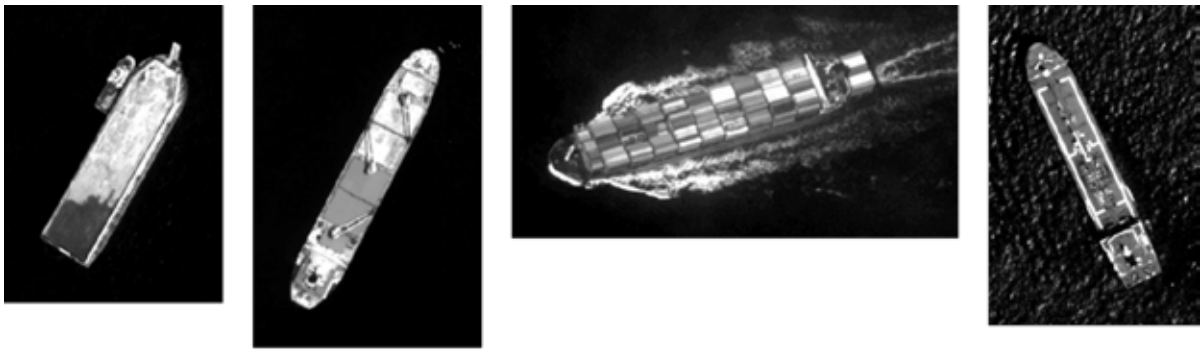


Figure 2. Original images. Source: [2].

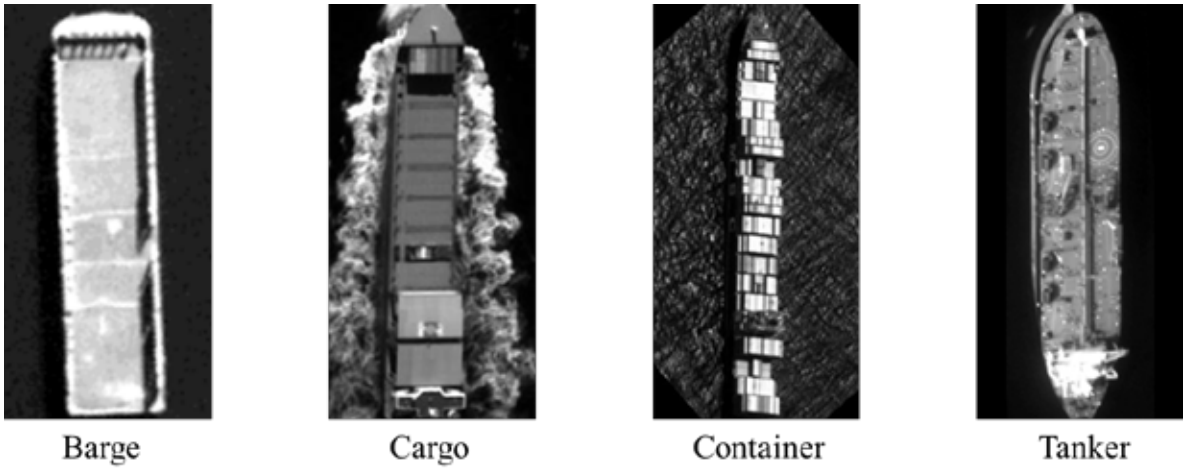


Figure 3. Resized images. Source: [2].

The quality of the image dataset has great influence on the effectiveness of the classifier algorithm [6], so it is of great importance to understand the limitations of BCCT-200. Image metadata such as sensor type, location, resolution, and sensor geometry were not kept when the dataset was compiled [6]. The vessel types were assigned by a layman with limited review [6]. BCCT-200 is a balanced dataset, with equal numbers of images for each class and a relatively large number of samples, but military interest is often in rare vessel classes [6]. However, BCCT-200 has served as a common benchmark for various ship classification algorithms and techniques.

B. PREVIOUS WORK

Rainey et al. conducted a broad survey to study the efficacy of several feature extractor algorithms and classifiers when applied to BCCT-200 [2]. The feature extractors used with resized images included Principal Component Analysis (PCA), Random Projection, Hierarchical Multiscale Local Binary Patterns (HMLBP), and Histogram of Oriented Gradients (HOG) [2]. The classifiers used were support vector machine (SVM), Sparse Representation-based Classification (SRC), and Nearest neighbor [2]. The only feature representation that was applied to the original images was visual bag of words (VBOW) clustered from scale invariant feature transform (SIFT) feature descriptors [2]. Parameswaran and Rainey conducted a follow-on study focused on the effect of term weighting to the VBOW due to evidence of classification performance improvement in the natural language processing field [3]. The VBOWs were built using the original images and several weights were applied to VBOWs of varying vocabulary sizes prior to being fed into the SVM classifier [3]. Rainey, Reeder and Corelli then studied the efficacy of a convolutional neural networks (CNN) in identifying a single class of ship among images of both various ship classes and non-ship images such as clouds and glints [7].

Two studies showed the performance improvement achieved through the Multiple Features Learning (MFL) framework when applied to the BCCT-200 resized and original images. The MFL combines features extracted from different algorithms to capture both local and global features. Huang et al. used three types of features: Gabor-based Multiscale Completed Local Binary Pattern (MS-CLBP), patch-based MS-CLBP and Fisher vector, and Spatial Pyramid Matching (SPM) augmented VBOW [8]. Shi et al. used a MFL of two-dimensional discrete fractional Fourier transform, Completed Local Binary Pattern (CLBP), and Gabor filter [9]. The MFL was fed into a deep CNN for classification [9].

Two studies have shown the efficacy of the random forest for vessel classification using Automated Information System data as features. Zhong, Song and Yang used vessel perimeter, area, ratio, and shape to classify cargo ships, tankers, and fishing vessels [10]. Snapir, Waine and Biermann used the vessel longitude, latitude, length, closest distance to shore, and time to distinguish between fishing and non-fishing vessels [11].

In Chapter II we introduce the theory and methodology of the feature extractors and classifiers used in this research. Next, in Chapter III, we discuss the parameters and implementation of the experiments. In Chapter IV we report the experimental results and in Chapter V we present our conclusions and recommendations for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

Image classification performances heavily rely on class features best suited to separate classes and the classification models considered in the classification tasks. In this chapter, we first introduce the feature extractors, HOG and LBP, used on the ship image database considered in the study [1]. Next, we introduce the visual bag of words (VBOW) and the algorithms selected to populate the vocabulary of visual words, the scale invariant feature transform (SIFT) in [3] and the speeded-up robust feature (SURF) in this study. Finally, we present a brief overview of the decision tree, random forest, and support vector machine models considered in the study.

A. FEATURE EXTRACTORS AND DESCRIPTORS

1. Histogram of Oriented Gradients

Histogram of oriented gradients is an edge detector, capturing the distribution of local intensity gradients in an image [12]. It was originally designed to facilitate human detection [12], and has since been used in a wide variety of image classification subjects. The process of generating the HOG feature vector is as follows.

The image is divided into cells of a given size. The horizontal (g_x) and vertical (g_y) gradients are determined by filtering the image with 1-dimensional centered derivative masks $[-1,0,1]$ and $[-1,0,1]^T$ [12]. The gradient magnitude (g) and direction (q) are calculated respectively as

$$g = \sqrt{g_x^2 + g_y^2} \tag{1}$$

and

$$q = \arctan(\frac{g_y}{g_x}). \tag{2}$$

If a gradient direction is negative, 180° is added or subtracted, resulting in unsigned gradients. In each cell, the magnitude of the unsigned gradients are then sorted by direction

in 9 bins ranging from 0°–180° to form a 9 x 1 histogram [13]. Note that the gradient magnitude is distributed between bins by interpolation when the gradient direction lies between two bins. The binning and interpolation process of an 8x8 cell is illustrated in Figure 4.

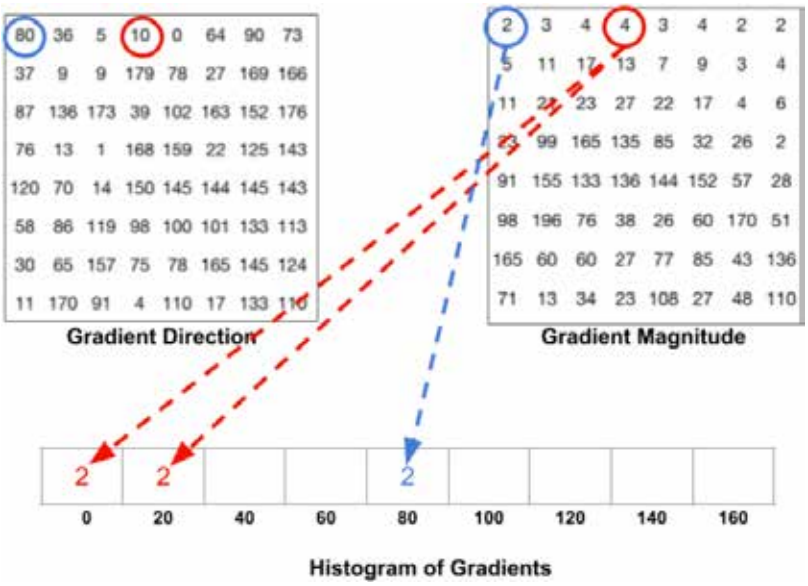


Figure 4. Creating histogram of 8 x 8 cell. Source: [13].

An overlay of the histograms in polar form [14] obtained from an image of a container ship is illustrated in Figure 5. The angle of the lines depicted in the right-hand image correspond to the bin and the line-length corresponds to the sum of gradient magnitudes.

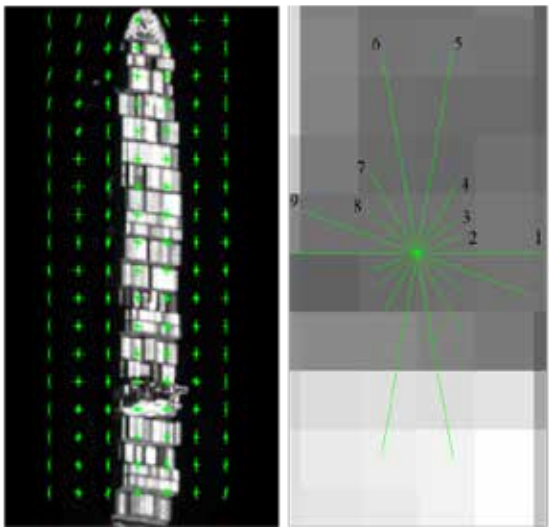


Figure 5. 9-bins of histograms illustrated and labeled

The image is then divided into blocks of 2 x 2 cells, which overlap each other by 2 cells horizontally and vertically. The four histograms in each block are concatenated to form a 36 x 1 vector. The 36 x 1 vector is then normalized by dividing every element by the L_2 norm of the vector [12]. The normalization takes into account local changes in illumination and contrast in the image [12]. The normalized 36 x 1 vectors of each block are then concatenated row by row to form the final feature vector of the image [13].

2. Local Binary Pattern

The local binary pattern was designed to capture local textures of an image [15], i.e., the spatial arrangement of pixel intensities [16]. LBP has shown outstanding results in facial analysis, and is useful in applications such as video conferencing and tracking and identifying people [17]. The process of generating the LBP feature vector is as follows.

The image is divided into cells of a given size, and in each cell, the grayscale intensity of every pixel is compared to the intensities of surrounding pixels. The sample point (g_p) is assigned 1 if its grayscale value is greater than the center pixel (g_o), zero otherwise [18]. The number of sampling points P on a circle of radius R for each comparison are selected by the user. The gray value is interpolated from surrounding pixels

if the point is not in a center of a pixel [15]. The comparison yields a cyclical binary code, as illustrated in Figure 6.

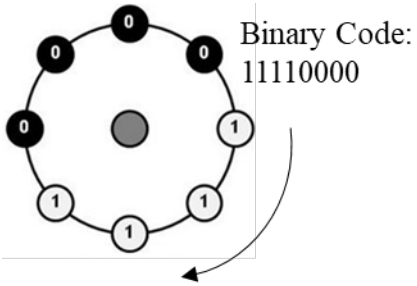
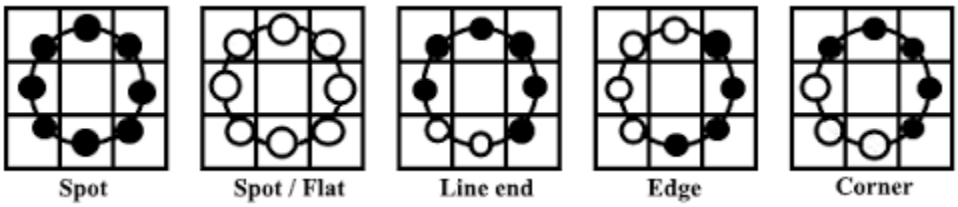


Figure 6. 8-bit binary code for center pixel. Adapted from [19].

In this feature descriptor, binary codes are classified as either uniform or non-uniform. The binary code for a pixel is considered uniform if there are two or less bit transitions, non-uniform otherwise [20]. In [21], it was shown that the uniform binary codes correspond to distinguishing features in images, such as spots, edges, line ends, and corners, as depicted in Figure 7.



Black and white circles correspond to bit values of 0 and 1 respectively.

Figure 7. Uniform patterns and associated features. Source: [22].

The binary codes are then converted to decimal values. A histogram of the decimal values is computed for each cell. To keep only distinguishing features, all non-uniform codes are grouped into one bin, regardless of decimal value. The number of bins in each histogram is calculated in [23] as

$$P(P - 1) + 3. \tag{3}$$

Each histogram is then normalized by its L_2 norm and are then concatenated row by row to form the final feature vector.

3. Scale Invariant Feature Transform

The scale invariant feature transform is widely used in image processing for object recognition and classification, biometrics, and robotics [24]. It is designed to find and describe interest points at different levels of resolution, or scales (σ), of the image [25]. These interest points are the locations of blobs, points or areas that stand out from their surrounding. The scales are composed of different octaves (doubling of σ) of the image, where the image of each successive octave is down-sampled by 2 to reduce computation [26]. Within each octave, the scale space $(L(x, y, \sigma))$ is calculated by convolving the image $(I(x, y))$ with Gaussian filters $(G(x, y, \sigma))$ of increasing scale, as shown in [26] as:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \tag{4}$$

where

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\left(\frac{-(x^2+y^2)}{2\sigma^2}\right)}. \tag{5}$$

Interest point locations correspond to local extrema of difference of Gaussian (DoG) blurred images $(D(x, y, \sigma))$, given in [26] as:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma), \tag{6}$$

where k is a constant multiplicative factor applied to the scale parameter. The Difference of Gaussian (DoG) function is a good approximation for the scale-normalized Laplacian of Gaussian, $\sigma^2 \nabla^2 G$, which results in scale-invariant interest points [26]. As illustrated in Figure 8, the convolution with Gaussians produce the scale space images on the left [26]. Adjacent Gaussian images in each octave are subtracted to produce the DoG images on the right.

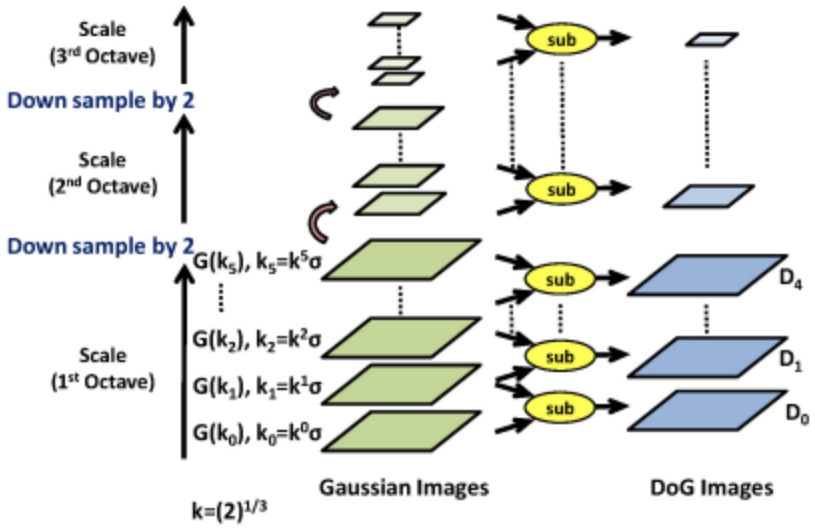


Figure 8. Gaussian and Difference of Gaussian (DoG) images. Source: [27].

Each pixel in the DoG images is compared to its 8 neighbors at the same scale and 9 neighbors on each of the adjacent scales [27]. Local maxima or minima are potential interest points. Low contrast candidates and responses along edges are removed because of their sensitivity to noise [26]. To assign an interest point its orientation, a gradient orientation histogram of 36 bins from 0°- 360° is populated from its neighborhood in the corresponding Gaussian filtered image [27]. Then the most frequently occurring orientation is assigned to the interest point [26]. The interest point descriptor is generated by dividing the neighborhood region centered on the key point in 16 square sub-regions. An 8-bin histogram of gradient orientations from 0° - 360° is populated for each sub-region, which are then concatenated to form a 128-dimensional feature vector. Rotation invariance is included by rotating the gradient orientations of the descriptor by the orientation of the interest point, and contrast invariance is achieved by dividing every element of the vector by the L_2 norm of the vector [26].

4. Speeded-Up Robust Feature

The speeded-up robust feature is scale and contrast invariant, similar to SIFT [28]. SURF improvements over SIFT include reduced sensitivity to noise and faster computational speed [28]. Interest points are detected by first calculating approximate

Hessian matrices at various scales. For a point (x, y) at scale s, the Hessian matrix is given in [28] as

$$H(x, y, s) = \begin{pmatrix} L_{xx}(x, y, s) & L_{xy}(x, y, s) \\ L_{xy}(x, y, s) & L_{yy}(x, y, s) \end{pmatrix} \tag{7}$$

where

$$L_{xx}(x, y, s) = \frac{\partial^2 G(x, y, s)}{\partial x^2} * I(x, y) \tag{8}$$

and similarly for $L_{xy}(x, y, s)$ and $L_{yy}(x, y, s)$. Note that the second order Gaussian derivative convolutions are approximated with box filter convolutions, similar to what is done in SIFT when the Laplacian of Gaussian is approximated. The convolution results are labeled with D_{xx}, D_{yy} and D_{xy}, as illustrated in Figure 9.

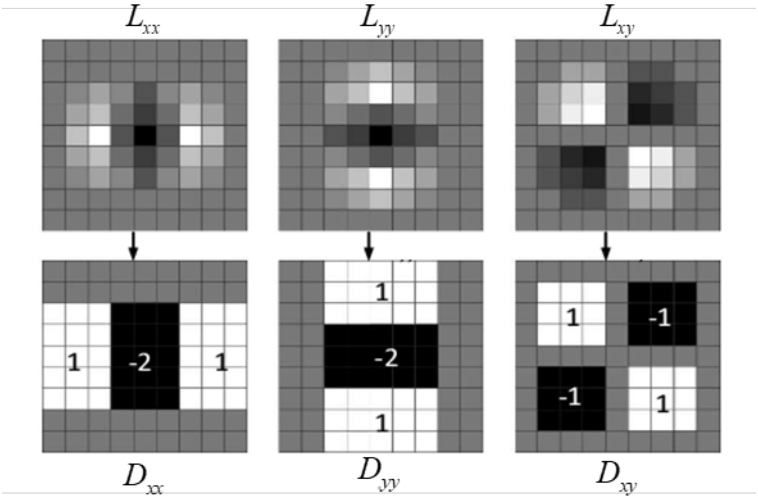


Figure 9. 9 x 9 discrete Gaussian second order partial derivative convolutions and corresponding box filter convolutions. Adapted from [29].

The convolution of the image and the box filter is computationally efficient due to use of integral images. The integral image $I(x, y)_{sum}$ obtained at any point (x, y) is the sum of all

pixel intensities above and to the left of that point [28]. The area of summation for the integral image of point D is illustrated as a blue rectangle in Figure 10. Once the integral image for each point in the image is calculated, it only takes a maximum of three operations to find the sum of intensities for a rectangle of any size. The sum of intensities of the gray rectangle (S) shown in Figure 10 is calculated using points A, B, C and D as

$$S = I(A)_{sum} - I(B)_{sum} - I(C)_{sum} + I(D)_{sum}. \tag{9}$$

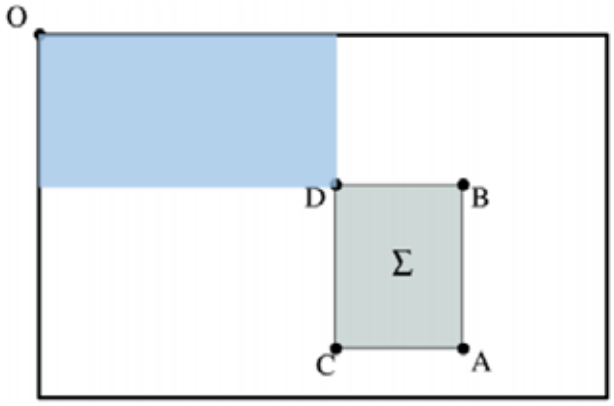


Figure 10. Integral image calculation. Adapted from [28].

The determinant of the resulting approximate Hessian is a blob detector, calculated in [28] as

$$\det(H_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2, \tag{10}$$

where the weight value w ; 0.9, is selected to conserve energy between the Gaussian second order derivatives and the box filters [28]. Box filter size is increased in octaves to find interest points at different scales, and then local maxima interest points are localised and interpolated in both the $x, y,$ and s dimensions. For each new octave, both the interval between subsequent filter sizes and sampling interval of interest points is doubled, as shown in Table 1, which reduces computation time [28]. The SURF descriptor is composed by finding the dominant orientation of gradients from a circle centered on the interest point

and then overlaying a square divided into 4 x 4 sections along the orientation [30]. Then Gaussian weighted Haar wavelet responses in the horizontal direction (d_x) and vertical direction (d_y), with respect to the interest point orientation, are summed in each section [28]. Polarity of intensity change information is encoded by including $|d_x|$ and $|d_y|$, resulting in a 4-dimensional vector seen in [28] as

$$v = \left(\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y| \right). \tag{11}$$

A 64-dimensional vector descriptor is obtained by concatenating the 4-dimensional vectors of the 16 sections of the overlaid square. There is an upright version of SURF (U-SURF) that does not encode the dominant orientation of the interest point [28]. U-SURF is not rotation invariant but it is faster to compute [28].

Table 1. SURF octaves and corresponding filter sizes. Adapted from [28].

Octave #	Box Filter Sizes	Filter Interval
1	9x9, 15x15, 21x21, 27x27	6
2	15x15, 27x27, 39x39, 51x51	12
3	27x27, 51x51, 75x75, 99x99	24
4	51x51, 99x99, 147x147, 195x195	48

An illustration of a sample of SURF interest points and histogram of those interest points are depicted in Figure 11 and Figure 12 respectively. The size of the circles centered on interest points shown in Figure 11 is directly proportional to the scale at which the interest point was detected. The histogram graphed in Figure 12 shows that the number of interest points rapidly decreases as scale increases.

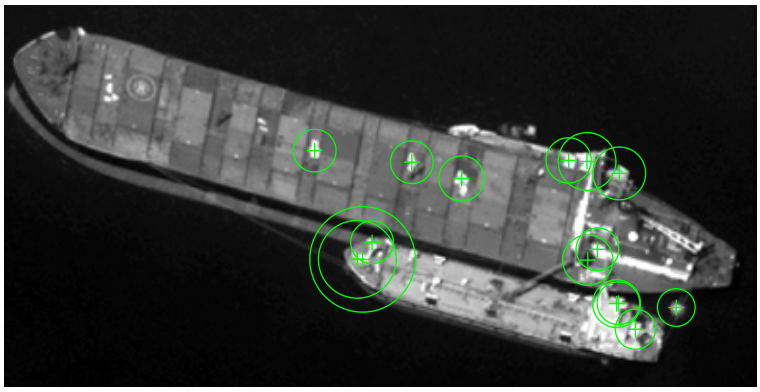


Figure 11. SURF interest points

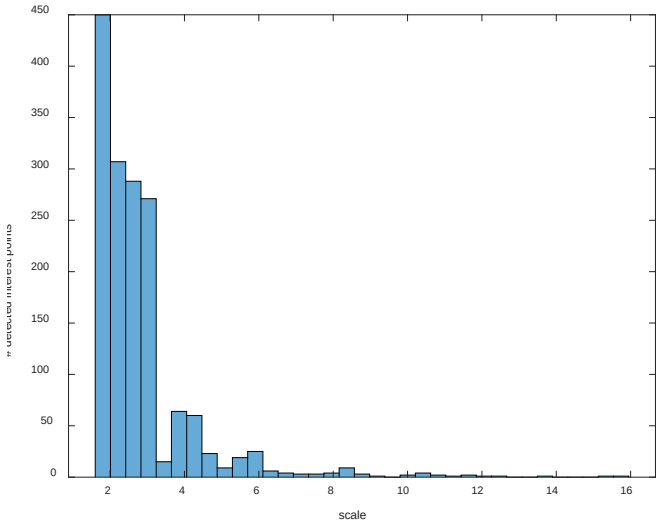


Figure 12. Histogram of interest points obtained from image in Figure 11

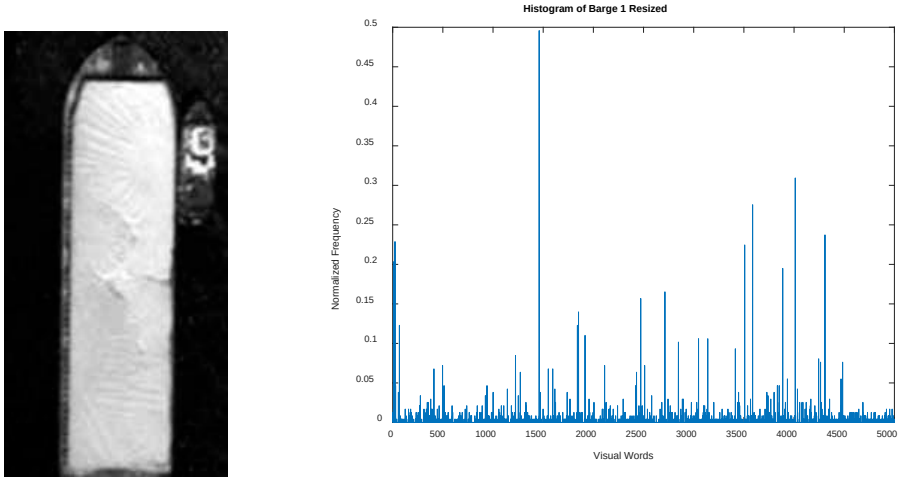
5. K-means Clustering

Classifiers such as SVM and random forest require all input feature vectors to be the same length [31]. However, the number of SURF interest points extracted from an image is image dependent. To fix the feature length, SURF interest points are combined by vector quantization [31]. First, SURF interest points are extracted from all images in the dataset. Then they are clustered using K-means clustering which labels each interest point by the index of the cluster it is assigned to [31]. The K-means algorithm starts with a group of k randomly selected points [32], i.e., cluster centers in the 64-dimensional feature space. Each data point is assigned to the nearest center. Each cluster center location is then

recomputed as the mean of the squared Euclidean distances of all points assigned to it [32]. Next, points are reassigned based off the updated locations of the cluster centers. This process is repeated to minimize the sum of squared Euclidean distances between points (x_i) and their nearest cluster center (k), given in [32] as

$$D(X,M) = \sum_{\text{cluster } k} \sum_{\substack{\text{point } i \text{ in} \\ \text{cluster } k}} (x_i - m_k)^2 . \tag{12}$$

Each cluster is considered a visual word, so k represents the chosen vocabulary size. Images are then represented by a histogram of frequencies of occurrence of the visual words, without regard to spatial arrangement, as illustrated in Figure 13. Note that the clustering process is the same for SIFT interest points. The difference is that the clustering takes place in a 128-dimensional feature space.



(a) (b)

Figure 13. (a) Barge 1 resized image and (b) 5000-word histogram

6. Visual Bag of Words

The concept of the visual bag of words is derived from the bag of words representation, which was developed to facilitate text classification [33]. The bag of words represents a document by a vector of the frequencies of occurrence of the words that

comprise the text [34]. Term weighting schemes applied to bag of words have shown to be effective in increasing text classification accuracy, and can also be applied to VBOW features to increase image classification accuracy [3].

Term weighting is composed of three factors: term frequency (TF) factor, collection frequency (CF) factor, and normalizing factor [3]. The term frequency factor depends on the frequency of occurrence of a visual word in an image [3]. The three TF factors used in this work are

1. Raw term frequency (RAW): simply the original representation [3]
2. Binary (BINARY) term frequency: assigns a 1 if the visual word is present and 0 if absent [3]
3. Log term frequency (LOG) weight is defined in [3] as

$$\log tf = \log(1 + tf). \tag{13}$$

The collection frequency factor includes both unsupervised and supervised weights. Unsupervised weights are not calculated with respect to the class of the vessel the visual word occurs in but supervised weights are class dependent [3]. The nine CF factors considered are

1. (Unsupervised) Inverse document frequency (IDF) weighs visual words by a factor inversely proportional to the number of images it appears in, calculated in [3] as

$$idf = \log(\frac{N}{n_w}), \tag{14}$$

where N is the total number of images and n_w is the number of images that contain visual word w.

2. (Unsupervised) Probabilistic inverse document frequency (PIDF) is a variant of IDF, described in [3] as

$$pidf = \log(\frac{N - n_w}{n_w}). \quad (15)$$

For the following equations, note N represents the total number of images, a and b are the number of images of vessel type i in which visual word t occurs and is absent respectively, and c and d are the number of images not of vessel type i in which t occurs and is absent respectively.

3. (Supervised) c^2 (CHI2) statistic assesses the level of dependence of visual word t and vessel type i and is given in [3] as

$$c^2(t,i) = \frac{N(ad - bc)^2}{(a+c)(b+d)(a+b)(c+d)}. \quad (16)$$

4. (Supervised) Mutual information (MI) measures the amount of information visual word t has about vessel type i and is estimated in [3] as

$$mi(t,i) = \log(\frac{aN}{(a+c)(a+b)}). \quad (17)$$

5. (Supervised) Information gain (IG) measures the reduction in entropy in a given vessel type if visual word t is present and it is expressed in [3] as

$$ig(t,i) = \frac{a}{N}' \log(\frac{aN}{(a+c)(a+b)}) + \frac{b}{N}' \log(\frac{bN}{(b+d)(a+b)}) + \frac{c}{N}' \log(\frac{cN}{(a+c)(c+d)}) + \frac{d}{N}' \log(\frac{dN}{(b+d)(c+d)}). \quad (18)$$

6. (Supervised) Odds Ratio (OR) describes the association between two values and is calculated in [3] as

$$or(t,i) = \frac{ad}{bc}. \quad (19)$$

7. (Supervised) Log Odds Ratio (LOGOR) takes the logarithm of Odds Ratio [3].

8. (Supervised) Relevance frequency (RF) weighs visual words according to the ratio of the number of images of vessel type i that do and do not contain visual word t, given in [3] as

$$rf(t,i) = \log(2 + \frac{a}{\max(1,c)}). \tag{20}$$

9. Unit weights (NONE) are applied for comparison.

Normalization is used to account for differences in image sizes [3]. Two techniques are considered:

1. No normalization.

2. L_2-normalized TF weights. TF factors are normalized before CF factors are applied [3].

B. CLASSIFIERS

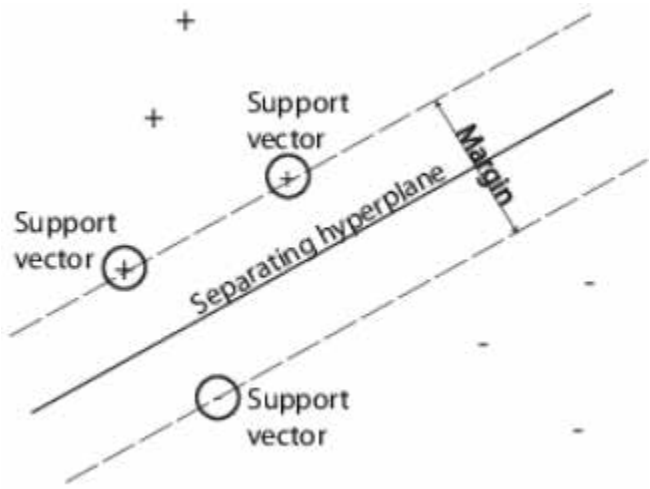
In this section, the two classifiers used in this research, support vector machines and random forest are introduced and compared. The discussion includes general theory and specific application of both in this research. A section on decision trees is also included to provide the foundation for understanding the random forest algorithm.

1. Support Vector Machine

The support vector machine (SVM) algorithm was originally designed for classification of binary data sets [35]. Common applications include image classification, biometrics, and text categorization [36]. In contrast to the random forest, SVMs have numerous hyper-parameters, need modification to process multi-class data sets, and require cross-validation to accurately assess performance.

To train an SVM, the data is first separated into a training set and a test set, in which the two classes are labelled 1 or -1. The SVM then generates a hyperplane that separates

the set of positive data from the set of negative data [37]. The position of the hyperplane is calculated to maximize the gap (margin) between the two classes by solving a constrained quadratic optimization problem [35]. The support vectors are the subset of training samples closest to the hyperplane [38], as illustrated in Figure 14.



Positive signs and negative signs correspond to positive and negative data respectively.

Figure 14. Hyperplane and associated support vectors. Source: [38].

However, the training data may not be completely separable due to outliers, noise, or non-linear characteristics [35]. Soft margin SVMs allow some training samples to fall within the margin and penalize those incorrectly classified. The weight of incorrectly classification is called the cost and is selected by the user. The SVM can be extended to non-linear models by mapping the original space into a higher-dimensional feature space, specified by the user-chosen kernel [37]. Some common kernel options are linear, polynomial, exponential, and Gaussian kernels [35].

In this research, multiple binary soft margin SVMs were used in an error correcting output code (ECOC) model to classify the multi-class dataset. The six SVMs considered each had a unique encoding of the four vessel classes. For each SVM, one class was assigned as the positive data (1), one assigned as the negative data (-1), and the other two

were removed prior to training (0), as recorded in Table 2, where m_{ix} is the matrix element corresponding to class i and SVM x.

Table 2. ECOC matrix

	SVM 1	SVM 2	SVM 3	SVM 4	SVM 5	SVM 6
Barge	1	1	1	0	0	0
Cargo	-1	0	0	1	1	0
Container	0	-1	0	-1	0	1
Tanker	0	0	-1	0	-1	-1

During testing, each image in the test set is fed into all six SVMs. Next, each SVM than computes a classification score (s_j) for the testing image, which is the signed distance from the image feature vector to the hyperplane [39]. The sign of the score corresponds to the sign of the predicted class. Binary loss $(g(m_{ix}, s_j))$ is a class specific measure that uses classification score to determine how well SVM x classifies a testing image into the class [40]. The predicted class $(\hat{i})$ of the testing image is the class which yields the minimum average binary loss, calculated in [40] as

$$\hat{i} = \min_i \left(\frac{\sum_{x=1}^{6} |m_{ix}| g(m_{ix}, s_j)}{\sum_{i=1}^{6} |m_{ix}|} \right). \quad (21)$$

The ratio of matches between the predicted and actual class is the accuracy of the overall classifier.

2. Decision Trees

In classification problems, decision trees apply a sequence of tests to input features to assign a predicted class label [41]. The decision tree is similar to the SVM in that both require the user to split the data into a training set and a test set. The decision tree is built through recursive partitioning of the training set [42]. Unlike SVMs, decision trees can

natively process inputs with any combination of numerical, categorical, and ordinal features [43]. The major disadvantage of decision trees is that they may over-fit to the training dataset. A single decision tree allowed to fully grow will classify the training set with high accuracy, but may not be accurate when classifying the test set. This sensitivity or variance is due to noise or coincidental irregularities in the training set [44].

The technique used to build decision trees in the random forest is the Classification and Regression Tree (CART) algorithm [45]. CART trees are binary decision trees that are grown by splitting a node into two child nodes recursively, beginning with the root node that contains the whole training set [46]. The splitting criteria is based on Gini Impurity, which in this work is defined as the probability of incorrectly classifying a randomly chosen vessel image if it were randomly labeled one of the classes [47]. Each node has an associated weighted Gini Impurity, which is calculated in [47] as

$$G = \frac{n}{N} \sum_{i=1}^{C} p(i)(1 - p(i)), \tag{22}$$

where n is the number of images sorted into that node, N is the total number of images in the training set, C is the total number of vessel classes, and $p(i)$ is the probability of randomly selecting an image belonging to class i in that node. In the dataset used in this research, all features are numerical, so the values of every feature are sorted from smallest to largest. CART then uses a greedy algorithm for feature selection, in which every value of every feature is tested to determine the feature t and threshold value v which minimizes the sum of Gini Impurities of the two child nodes [48]. As a matter of convention, images with $t < v$ are sent to the left child node and $t \geq v$ are sent to the right. The splitting is continued until all images at a node belong to the same class $(G = 0)$ or if all images in a node have the same values for each feature [46]. These leaf nodes are labelled with the class of the majority. A decision tree with various features and thresholds is illustrated in Figure 15. The images in the test set are then run down the decision tree, and the output is the label of the leaf node they are sorted into. The output is compared to the true classification to determine the accuracy.

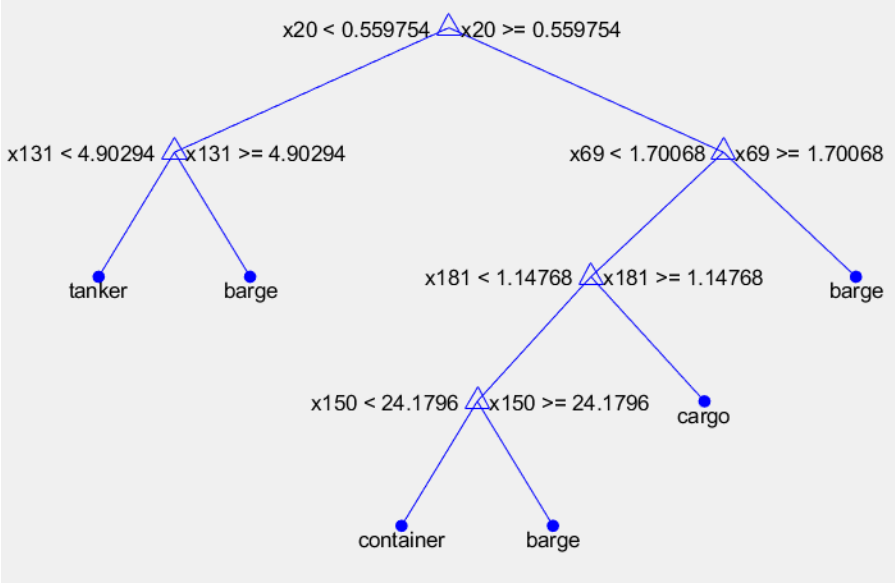


Figure 15. Decision tree

3. Random Forest

Random forest, developed by Breiman in 2001 [45], is an ensemble method applied to decision trees. From a specific feature set D, sets $D_1,...,D_M$ are populated by bootstrapping from D, i.e., selecting random samples with replacement. All sets have the same number of samples as D, but approximately one-third of the original data is left out of each bootstrapped set, which results in an "out-of-bag" data set [49]. For the feature set used in this research, the probability of a feature vector not being chosen for a dataset is calculated in [50] as

$$P(\text{not chosen}) = (1-\frac{1}{n})^n = (1-\frac{1}{800})^{800} \approx 0.368, \tag{23}$$

where n is the number of samples in the original feature set.

Then each bootstrapped sample is used to train a decision tree. Feature selection conducted in the random forest algorithm differs from that conducted in the regular decision tree algorithm. Instead of considering all possible features for assignment to the root and internal nodes as done in basic decision trees, only a random sub-sample of

features f are selected for consideration at each node [45]. That random subsampling of features is applied to reduce correlation among the trees, and results in higher overall accuracy for the random forest. Previous work has shown setting f to approximately the square root of the total number of features produces the best results [51]. Class decision for an image is obtained by selecting the class decision which occurs the most frequently out of all trees included in the forest. To classify a new image, the feature vector of the image is run down every tree of the forest [22]. The class output by each model is one vote and the class that receives the majority of votes is returned by the forest [23].

The simplicity of the random forest is that it requires only two inputs: the number of trees grown in the forest (M) and the number of features (f) considered at each node. Overall results have been found to be resistant to overfitting due to the bagging process used for the final decision step. In addition, unlike SVM, the data is not split by the user into a training set and a test set and cross-validation is not required. Each image in the "out-of-bag" dataset is run down the trees that did not use it in the tree construction stage, approximately one-third of the trees. The output class of the forest is compared to the true class, and the resulting out-of-bag error is the proportion of instances without match between true and predicted class assignments [22].

C. SUMMARY

In this chapter we first discussed the feature extractors and classifiers selected for this study. Note that the row-by-row concatenation of cell histograms to create the final histogram in the HOG and LBP method encodes spatial and structural information. However, the concatenation limits HOG and LBP to use on images of uniform size. Because the clustering of SURF/SIFT interest point descriptors omits spatial information, the VBOW feature representation can be applied to datasets with images of non-uniform size. Finally, we introduced the two types of classifiers considered in this study: SVM and random forest. In the next chapter, we describe the parameters and order of experimentation followed in this research.

THIS PAGE INTENTIONALLY LEFT BLANK

III. EXPERIMENTAL APPROACH

In this chapter, we describe the investigations conducted during the study to evaluate the performance of the random forest model in classifying the BCCT-200 ship images considered. The random forest classifier is first applied to resized images. The goal is to compare the performance of the random forest to the SVM classifier, using the HOG, LBP, and VBOW as feature extractors. Then variations to HOG, LBP, VBOW and the random forest itself are applied to optimize the accuracy of the random forest. Then an augmentation to the VBOW called Spatial Pyramid Matching is applied. The random forest classifier is then applied to the original images, using the VBOW feature representation. The experimental approach with original images largely follows the 2015 work by Parameswaran and Rainey [3], in which VBOW vocabulary size was varied and multiple term frequency and collection frequency weights were applied prior to being fed into an SVM classifier.

A. VISUAL BAG OF WORDS AND TERM WEIGHTING

SURF features are used to populate the VBOW. U-SURF is applied to resized images as they are aligned. The rotation-invariant SURF is applied to the original images. Simulations show U-SURF extracted 2,310,400 features from all resized images, and SURF extracted 4,033,540 features from all original images. The top 80% strongest features were kept to populate the VBOW. The strength is calculated as the magnitude of the scale-normalized (s) Laplacian of the intensity of the interest point, as presented in [52] as

$$Strength = \left| s\tilde{\nabla}^2 I(x, y, s) \right|. \tag{24}$$

The kept features are then clustered to form the given number of visual words using the k-means algorithm, as discussed earlier. In our work, we allow the k-means algorithm up to 100 iterations to converge, which was shown to be sufficient to reach convergence.

The final feature set is compiled by encoding each image as a histogram of visual words, and then weighted by given term frequency and collection frequency factors. However, the supervised CF factors (c^2, IG, MI, OR, LOGOR, RF) were originally defined for two-class datasets [3]. As a result, we selected an extension originally proposed in [33] to apply to our four-class scenario; first, weights are calculated for every feature per class, next, the final CF factor is compiled by taking the maximum of the four weights for each feature [3]. The two normalization schemes are also applied to the VBOW of original images with the RAW and LOG TF factors, yielding three sets of weighted features for resized images and five sets of weighted features for original images, as outlined in Table 3.

Table 3. VBOW datasets for resized and original images

Resized Images	Original Images
RAW TF + Various CF Weights	RAW TF + Various CF Weights
BINARY TF + Various CF Weights	BINARY TF + Various CF Weights
LOG TF + Various CF Weights	LOG TF + Various CF Weights
	L_2 Normed RAW TF + Various CF Weights
	L_2 Normed LOG TF + Various CF Weights

B. RANDOM FOREST CONSTRUCTION AND ANALYSIS

The out-of-bag error of a random forest is calculated concurrently with the growing of the forest. For example, there will be 100 out-of-bag error values for a 100-tree random forest, with each value providing the ensemble error of the forest grown to that point [53]. The out-of-bag error is calculated only with out-of-bag observations, which may result in large variances in error when only a small number of trees are grown due to the random nature of bootstrapping. Therefore, a large enough number of trees are grown to yield a stable out-of-bag error, as shown in Figure 16, which shows that error has stabilized for 3000 trees. For features extracted using HOG and LBP, a 3000-tree random forest was

grown. For VBOW features, 3000-tree and 1500-tree random forests were grown for un-supervised CF weighted VBOW and supervised CF weighted VBOW, respectively.

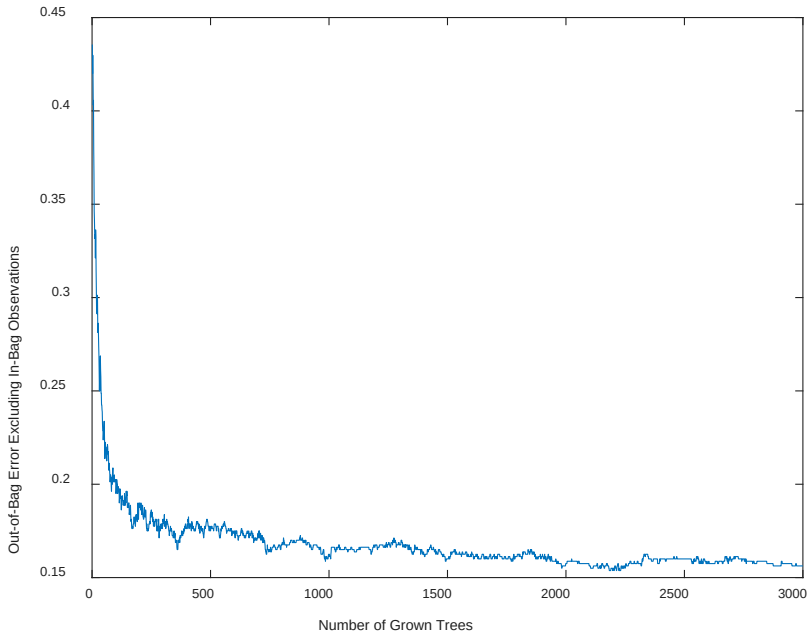


Figure 16. Out-of-bag error vs. number of trees

To consistently compare performance of each random forest, the out-of-bag error of the fully grown forest is used to report classification accuracy, calculated as

$$Accuracy(\%) = 100 * (1 - \text{(out-of-bag error)}). \quad (25)$$

Recall the out-of-bag error is based on classification predictions from only about one-third of the total trees grown [45]. Therefore, the out-of-bag error will tend to overestimate the true error rate [45], so there is little risk in overestimating the accuracy of the Random Forest.

C. COMPARISON OF SVM AND RANDOM FOREST PERFORMANCE ON RESIZED IMAGES

The random forest is compared to two SVM models, one run in this work and one in a previous work by Parameswaran and Rainey [3], both with an 80%–20% training/

testing divide averaged over five runs on the resized images [1]. The SVM model in [2] used a linear and Gaussian kernel, but the results did not specify the specific parameters selected to define these kernels. HOG, HMLBP, and a SIFT populated 500-word VBOW were used with the previous SVM model. The SVM model used in this work uses the MATLAB built-in linear kernel. HOG, LBP, and a SURF populated 500-word VBOW is used with Random Forest and the current SVM model. The number of features extracted with HOG and HMLBP in Rainey et al. [2] is unknown, so HOG and LBP parameters selected in our work are chosen to yield approximately the same number of features, as presented in Table 4. The number of features randomly sampled at each level of the random forest (f) is calculated as

$$f = \left\lfloor \sqrt{F} + 0.5 \right\rfloor, \tag{26}$$

where F is the total number of features.

Table 4. HOG, LBP and VBOW parameters

Feature Algorithm	Cell Size	Sample Points	Radius	No. of Features (F)	No. of sub-sampled Features (f)
HOG	16 x 16	N/A	N/A	4896	70
LBP	32 x 32	12	3	4860	70
VBOW	N/A	N/A	N/A	500	22

D. HOG AND LBP

The cell size of the HOG and LBP are varied to determine the number of features optimizing random forest accuracy, as shown in Table 5. The LBP number of sample points and radius values were kept constant.

Table 5. HOG and LBP parameters

HOG (Cell Size)	No. of Features (F)	No. of sub-sampled Features (f)	LBP (Cell Size)	No. of Features (F)	No. of sub-sampled Features (f)
8x8	22032	148	16x16	21870	148
12x12	9504	97	24x24	9720	99
16x16	4896	70	32x32	4860	70
20x20	3024	55	40x40	2835	53

E. APPLICATION OF VBOW AND RANDOM FOREST TO RESIZED IMAGES

1. Varying Word Size and Term Weights

RAW, BINARY, and LOG term frequency and all collection frequency weights were applied to a variety of vocabularies. Vocabulary sizes used were: 100, 200, 500, 1000, 5000, 10000 and 15000 words.

2. Spatial Pyramid Matching

In Huang et al., augmenting VBOW with Spatial Pyramid Matching was shown to increase classification accuracy for the resized images by adding structural and spatial information to the final histogram [8].

In SPM, the image is divided into sub-images of equal size 2^l x 2^l segments, where l is referred to as the layer number starting at 0. The segments in each layer are used to create a VBOW. Then each image is encoded with their corresponding layer VBOW to create their frequency histogram. In our study, we used 3 layers, corresponding to $l = 0,1,2.$ Finally, histograms from these three layers were concatenated together from the 0^{th} to 2^{nd} layer to create one overall histogram [8]. Note that the concatenation of the segment histograms and then the layer histograms happen in the same order for each final histogram. An illustration of a a three-layer pyramid is illustrated in Figure 17.

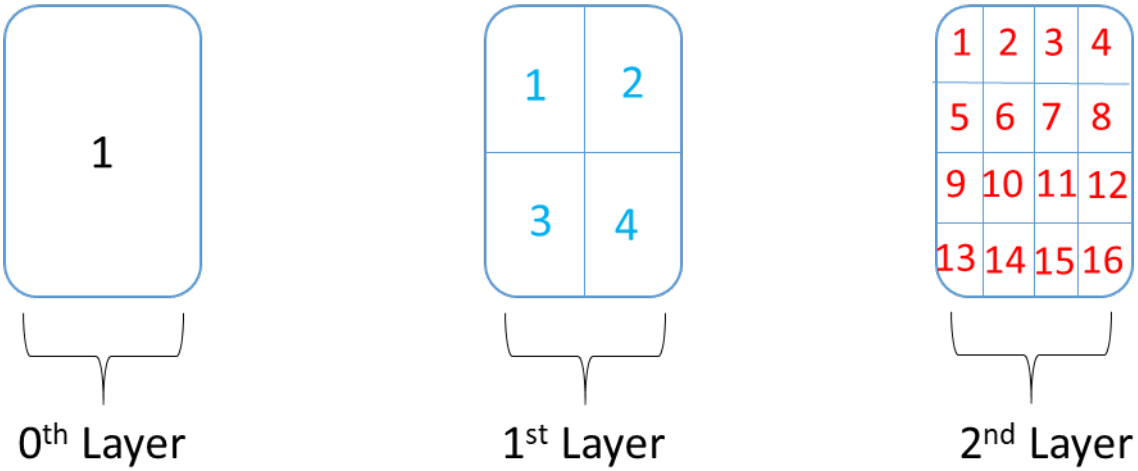


Figure 17. Illustration of three-level SPM.

We applied two variants of SPM in our study. For both, words per segment were calculated so that the total vocabulary size would be close to 5000 words and therefore results could be compared to the 5000-word VBOW without SPM applied. In the first, 238 words were extracted from the segments of each layer, the total number of segments was $1+4+16=21$. The resulting total number of words was $238(21)=4998$ words. In the second, the number of words per segment were proportional to the image size of the segment, calculated as

$$5000 \text{ words}/3 \text{ layers}/1 \text{ segment} \approx 1668 \text{ words/segment} \tag{27}$$

for the 0^{th} layer,

$$5000 \text{ words}/3 \text{ layers}/4 \text{ segments} \approx 417 \text{ words/segment} \tag{28}$$

for the 1^{st} layer, and

$$5000 \text{ words}/3 \text{ layers}/16 \text{ segments} \approx 104 \text{ words/segment} \tag{29}$$

for the 2^{nd} layer, totaling

$$\begin{aligned} &1668 \text{ words/segment}(1 \text{ segment}) + 417 \text{ words/segment}(4 \text{ segments}) + \\ &104 \text{ words/segment}(16 \text{ segments}) = 5000 \text{ words}. \end{aligned} \tag{30}$$

F. APPLICATION OF VBOW AND RANDOM FOREST ON ORIGINAL IMAGES

Experiments were conducted on VBOW features of the following vocabulary sizes: 100, 200, 500, 1000, 5000, 10000, and 15000. The RAW, BINARY, and LOG term frequencies weights and all collection frequency weights were applied. The smallest vocabulary size that does not reduce classification accuracy is desirable because it allows the decision trees to grow more quickly, thus reducing total computation time. The random forest performance is compared to that of the linear kernel SVM used in [3].

G. SUMMARY

This chapter outlined the methodology used in feature extraction and classification. When random forest is applied to resized images, our goal is to investigate the viability of the random forest as an image classifier by comparing its performance to that obtained from the SVM model. Once viability is shown, changes are made to the feature extractor parameters to enhance the random forest accuracy. Next, the practicality of the VBOW feature representation and term weights were assessed. Two comparisons are conducted: the first between the performance of the weighted VBOW on resized and original images when random forest is the classifier, and the second between SVM and random forest classification accuracy obtained on the original images.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. EXPERIMENTAL RESULTS

In this chapter the random forest and SVM classification accuracy obtained in the research is presented and analyzed. The order of presentation corresponds to the experimental order outlined in Chapter III.

A. RESIZED IMAGES

1. Random Forest and SVM

Results show the random forest performance exceeds or is similar to that of the SVM used in this study, as reflected in Table 6. The accuracy of both the SVM and random forest used in this work exceed that of the SVM in the previous work by approximately 10%. For HOG and LBP, the better performance may be due to the greater number of features extracted. For VBOW, the improved performance of SVM model used in this work may be due to differences in the SIFT and SURF features or different parameters in the classifiers themselves.

Table 6. Random forest and SVM accuracy

	Random forest Accuracy (%)	SVM Accuracy (%) (Current)	SVM Accuracy (%) (Previous) [6]
HOG	94.00	91.80	81.60
LBP/ HMLBP	91.25	89.80	90.80
VBOW	84.37	85.60	76.80

2. Varying No. of Features Extracted Using HOG and LBP

The highest performance achieved with HOG and LBP was found to be 94.00% and 93.25% respectively, as shown in Table 7. Results show the accuracy of the random forest is inversely proportional to the cell size for LBP. The size of the HOG feature set which yielded the best result is 4896, while the size of the LBP feature set which yielded the best results is 21870. Therefore, HOG may be the more effective feature extractor when used with the random forest classifier.

Table 7. Random forest accuracy as HOG and LBP cell size varies

HOG (Cell Size + No. of Features)	Accuracy (%)	LBP (Cell Size + No. of Features)	Accuracy (%)
8 x 8 (22032 Features)	92.00	16 x 16 (21870 Features)	93.25
12 x 12 (9504 Features)	93.25	24 x 24 (9720 Features)	91.75
16 x 16 (4896 Features)	94.00	32 x 32 (4860 Features)	91.25
20 x 20 (3024 Feature)	93.00	40 x 40 (2835 Features)	89.87

3. Varying Vocabulary Size

Figures 18–20 present random forest accuracy achieved when applying RAW, BINARY, and LOG TF weights in combination with all CF weights. Results show the RAW and LOG TF accuracies to be similar, both sharing minimal increases in accuracy as the vocabulary size increases. The BINARY TF results are the most responsive to vocabulary size increases, ranging from 52.50% for the 100-word vocabulary to 86.75% for the 15000-word vocabulary. The only instance when CF factors affected the accuracy for all three TF was for the 100-word VBOW, where OR and LOGOR CF weights reduced accuracy 3 - 5%. Elsewhere, their influence was shown to be minimal. The optimal vocabulary size selected for all three was 10000 and 15000 words, yielding similar accuracies for all three. These findings indicate that the random forest may be classifying images effectively based solely on the presence or absence of visual words, not on their frequency of occurrence in the image when the vocabulary size is sufficiently large.

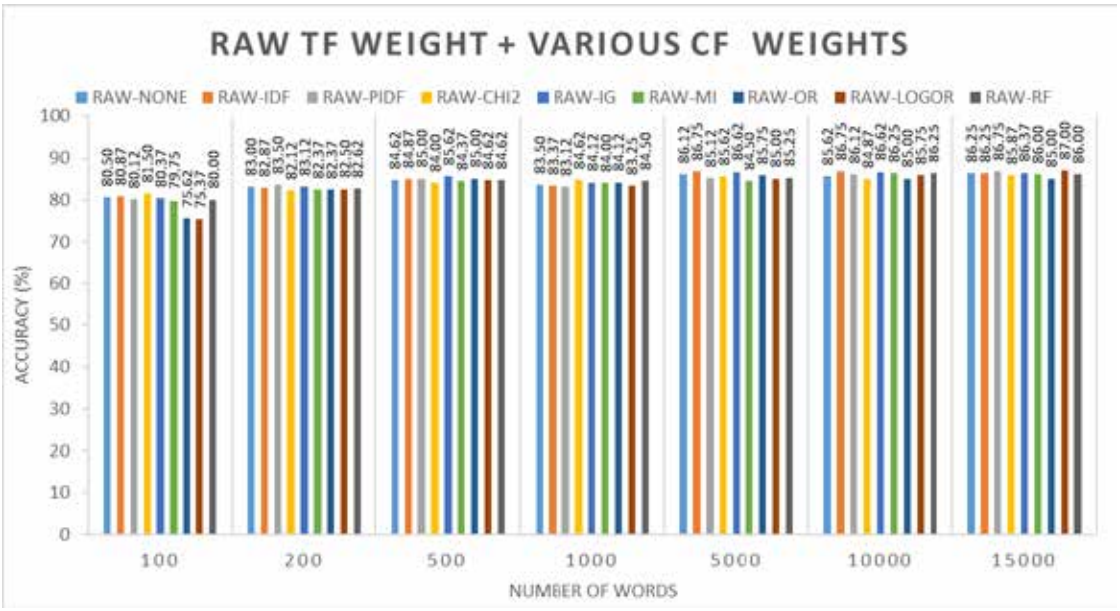


Figure 18. Random forest results with RAW TF and all CF weighted features

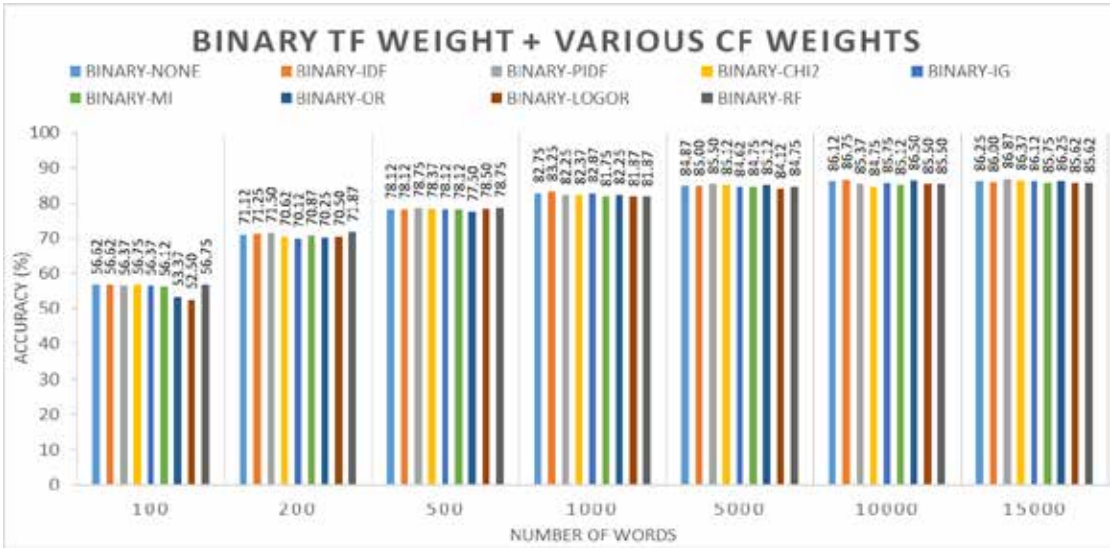


Figure 19. Random forest results with BINARY TF and all CF weighted features

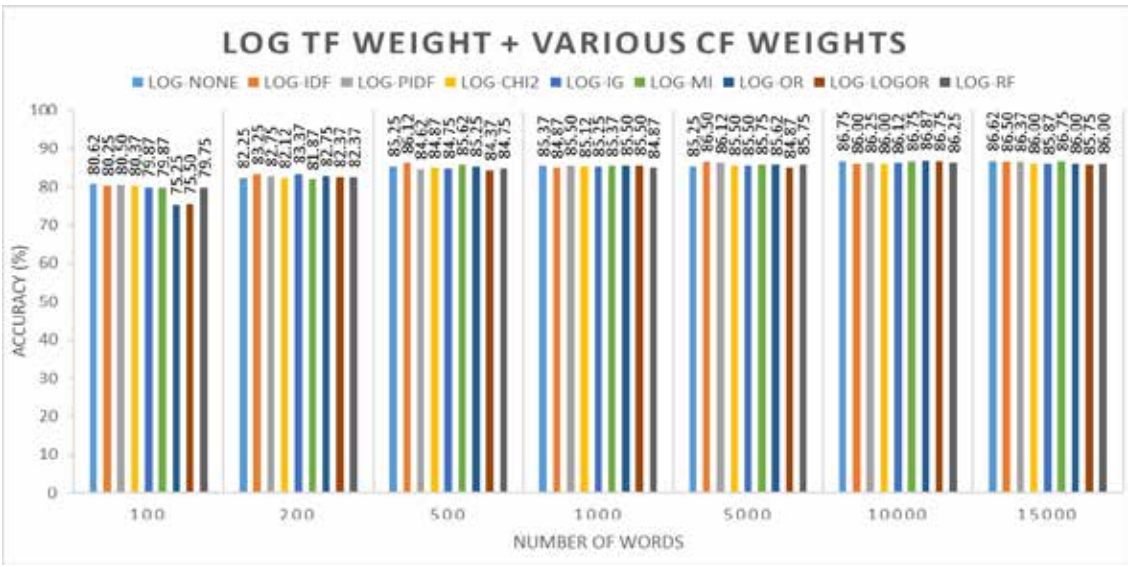


Figure 20. Random forest results with LOG TF and all CF weighted features

4. Spatial Pyramid Matching

Results show the addition of spatial information through the SPM consistently increased the accuracy of the random forest, as shown in Table 8. The accuracies increased anywhere from 1–3%. Both variants of the SPM considered in this study performed similarly.

Table 8. Random forest results with SPM and without SPM augmentation to 5000-word VBOW

	SPM (Proportional No. of Words/ Segment) Accuracy (%)	SPM (Equal No. of Words/Segment) Accuracy (%)	No SPM Accuracy (%)
RAW-NONE	88.00	88.87	86.12
RAW-IDF	88.00	87.62	86.75
RAW-PIDF	88.75	88.62	85.12
RAW-CHI2	88.50	88.00	85.62
RAW-IG	88.12	88.12	86.62
RAW-MI	88.12	88.25	84.50
RAW-OR	88.62	88.50	85.75
RAW-LOGOR	88.50	87.50	85.00
RAW-RF	88.25	87.50	85.25

B. ORIGINAL IMAGES

Figures 21, 23, and 25 show random forest results of all three TF weights, with no normalization and all CF weights applied. Figures 22, 24, and 26 show corresponding SVM results obtained in [3]. Results show the application of RAW and LOG TF weights resulted in much lower accuracies for the random forest than those obtained with the SVM. For the random forest, there was minimal change in accuracy level as vocabulary size varied, but 1000-word vocabularies yielded the highest accuracies. In contrast to the SVM results, random forest performance with all three TF weights was not affected by the application of any CF weights, un-supervised or supervised.

Simulations show that for the random forest, the BINARY TF weighted VBOW results are the most responsive to vocabulary size increases and yielded comparable results to the SVM for the large vocabulary sizes (10000 and 15000). The similar behavior of the BINARY TF weighted VBOW results for both resized and original images also provides further evidence that the random forest is effectively classifying images based solely on the presence or absence of visual words when the vocabulary size is large enough.

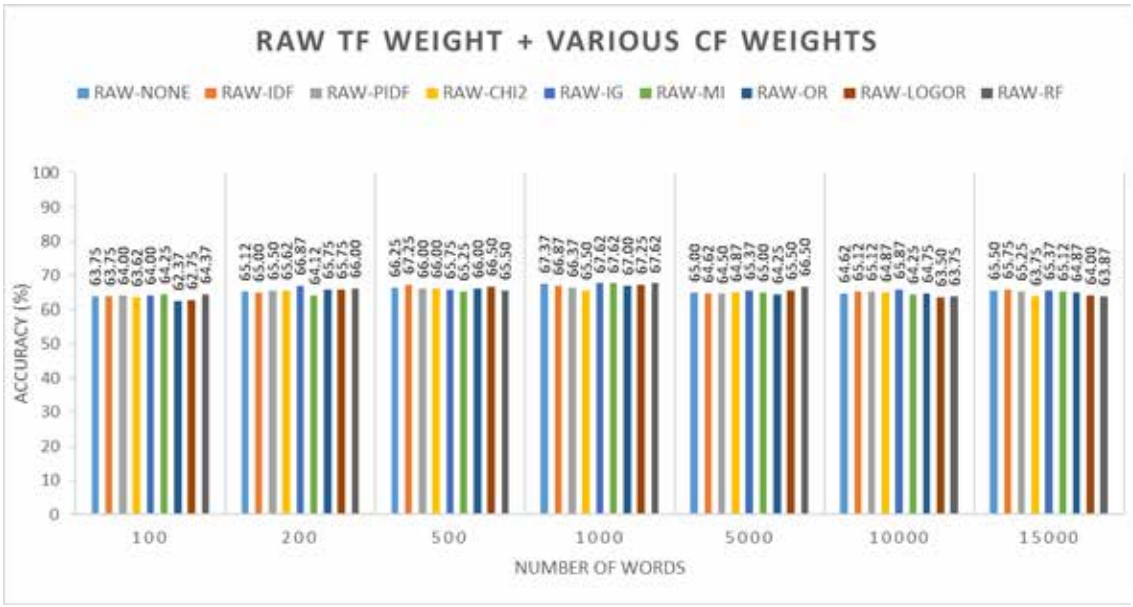


Figure 21. Random forest results with RAW TF and all CF weighted features

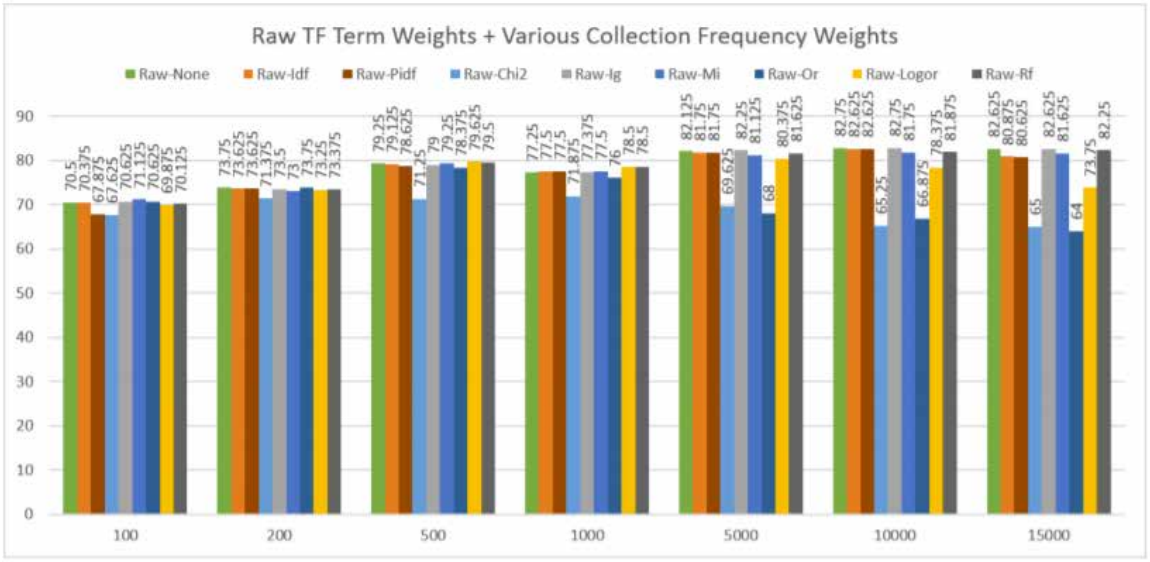


Figure 22. SVM results with RAW TF and all CF weighted features. Source: [3].

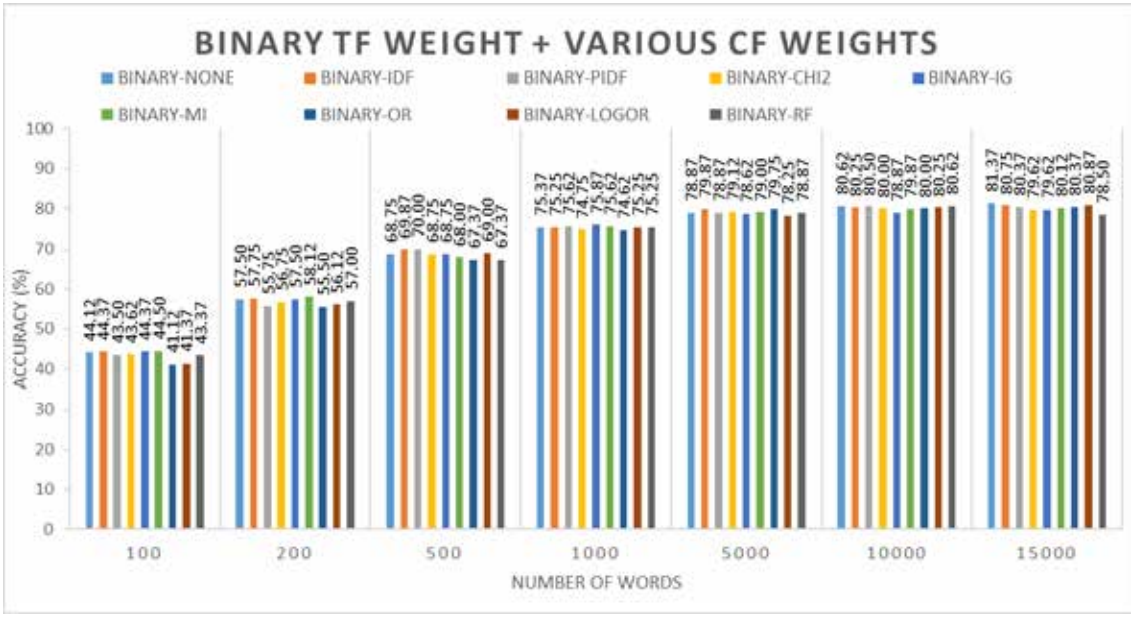


Figure 23. Random forest results with BINARY TF and all CF weighted features

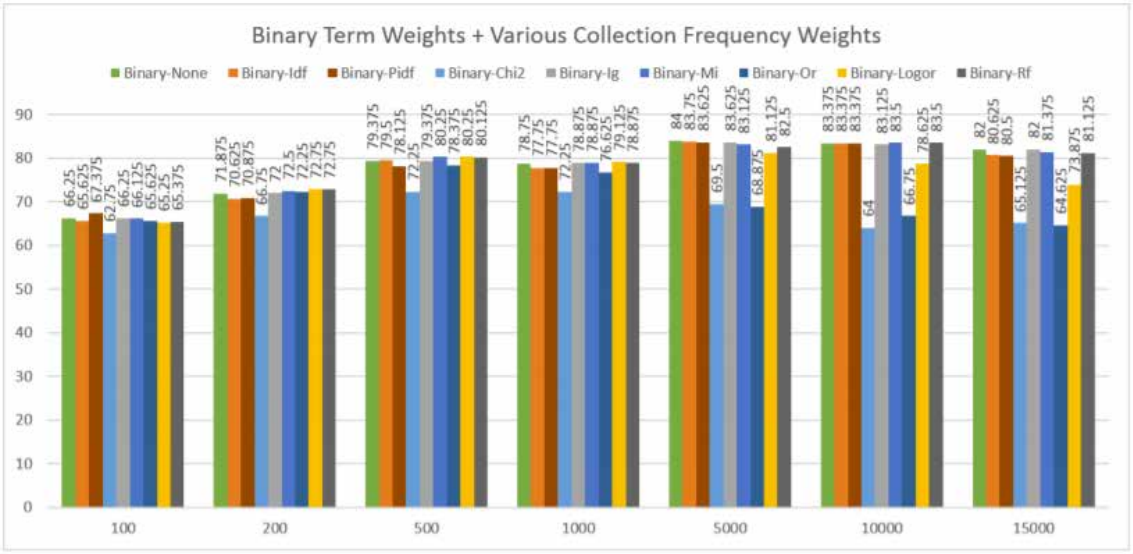


Figure 24. SVM results with BINARY TF and all CF weighted features. Source: [3].

Figure 25. Random forest results with LOG TF and all CF weighted features.

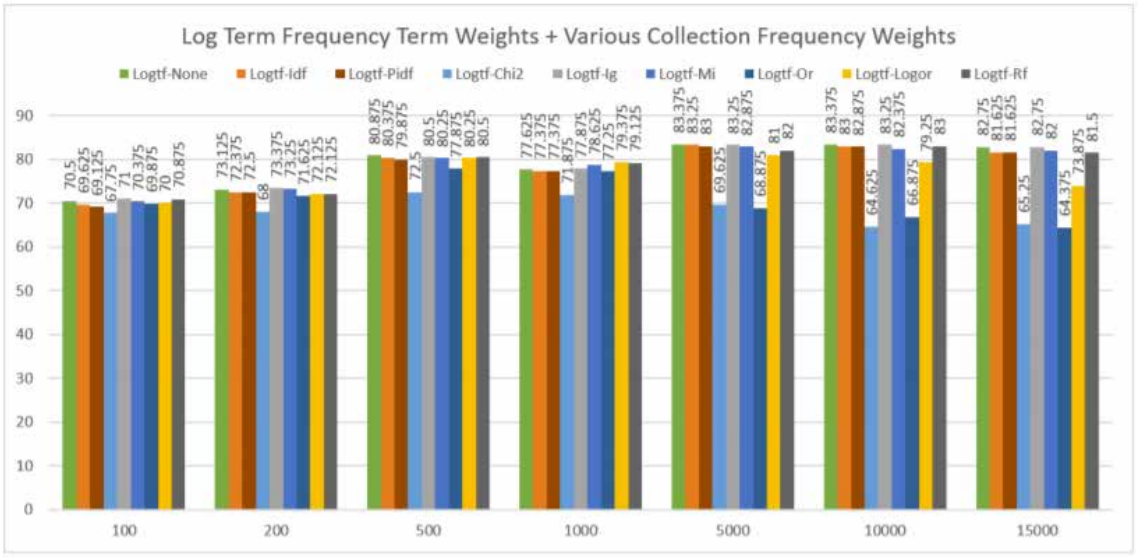


Figure 26. SVM results with LOG TF and all CF weighted features. Source: [3].

Figures 27 and 29 show classification performances obtained for the random forest classifier using the RAW and LOG TF weighted VBOW, with L_2 normalization and all CF weights applied. Figures 28 and 30 show corresponding SVM results obtained in [3]. Results show normalization has little effect on the random forest performance. The SVM clearly outperforms the random forest for all vocabulary sizes. The only case random forest outperforms the SVM is when the CHI2 CF weight is applied to the 10000 and 15000 word vocabularies. The unsupervised and supervised CF weights again have minimal effect on the random forest accuracy.

Figure 27. Random forest results with L_2 normed RAW TF and all CF weighted features

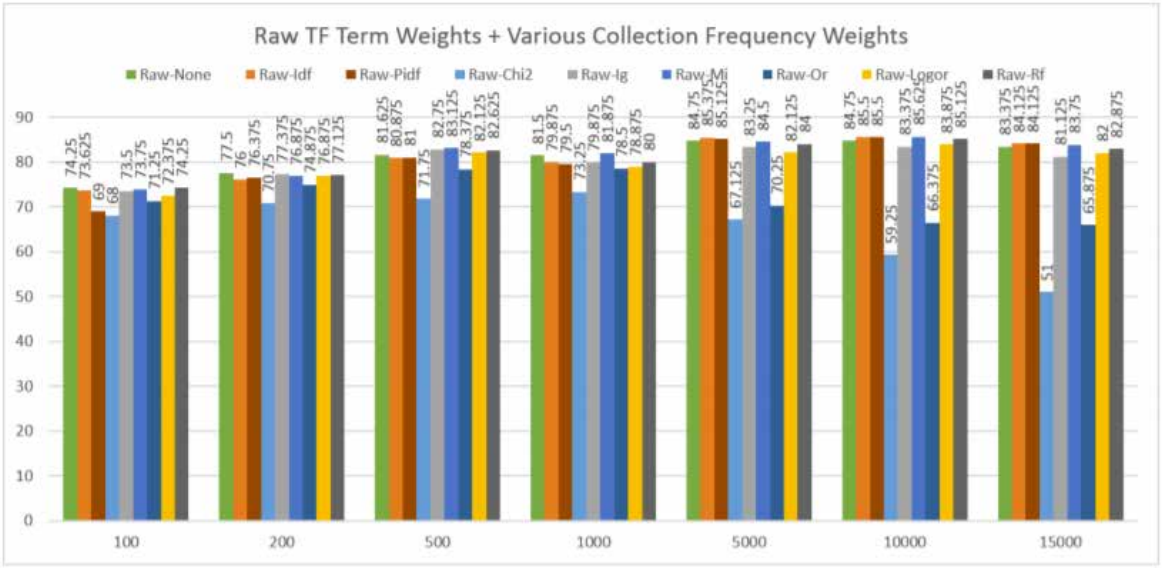


Figure 28. SVM results with L_2 normed RAW TF and all CF weighted features. Source: [3].

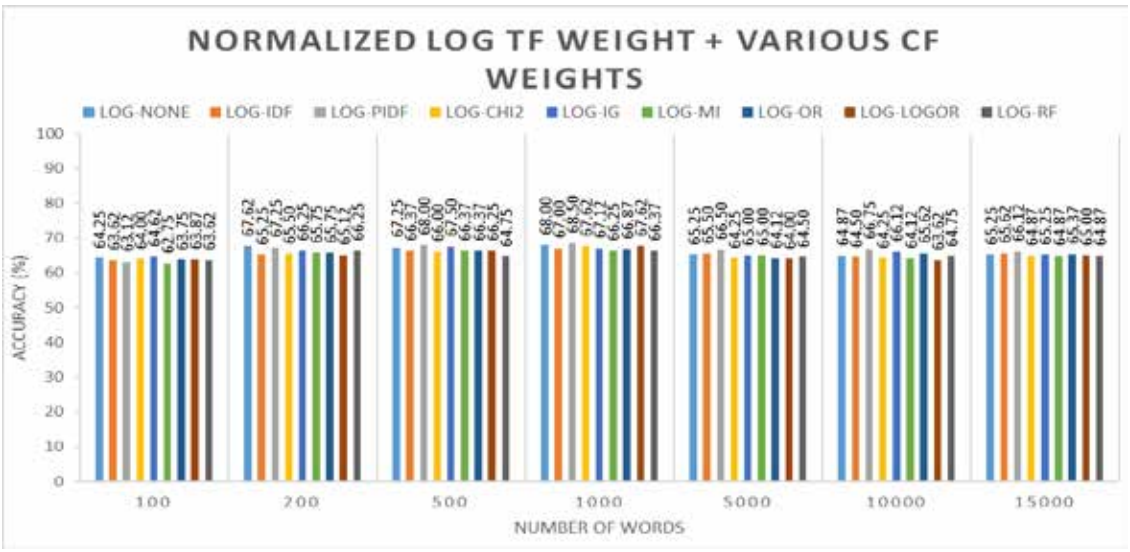


Figure 29. Random forest results with L_2 normed LOG TF and all CF weighted features

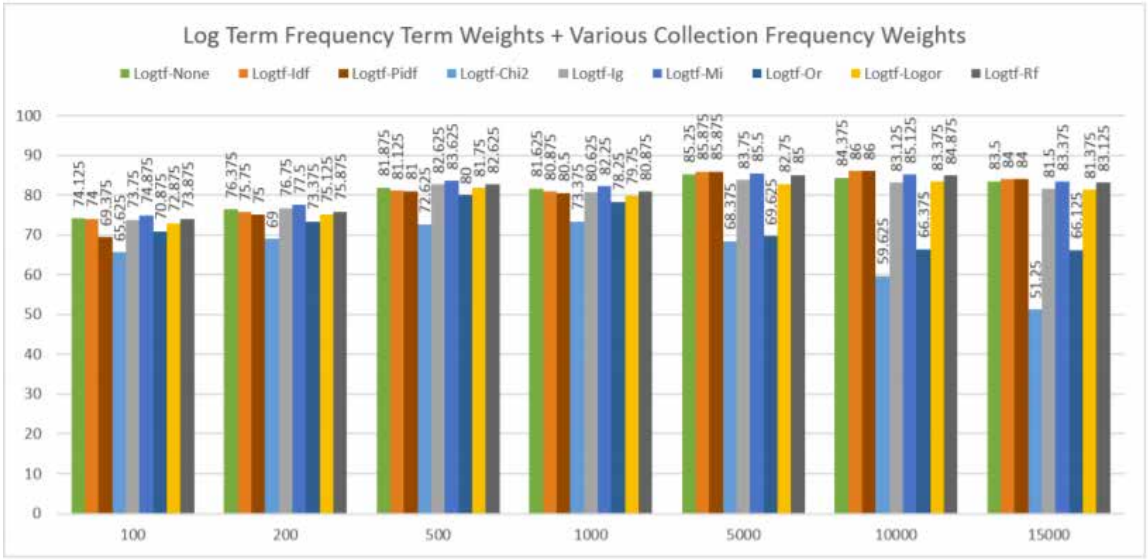


Figure 30. SVM results with L_2 normed LOG TF and all CF weighted features. Source: [3].

C. SUMMARY

In this chapter we presented the classification accuracies obtained from the SVM and random forest classifier when applied to the resized and original images. The random forest outperformed the SVM when applied to resized images but greatly underperformed

when applied to the original images. We discuss conclusions drawn from these results and recommendations for future work in the next chapter.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSIONS AND RECOMMENDATIONS

Due to the greater availability of satellite imagery, ship detection and classification is an increasingly popular area of study [1]. Contributing to that body of work, we investigated the effectiveness of the random forest algorithm in conjunction with various feature extractors in classifying vessel types. The random forest ease of use and computational efficiency make it an attractive classifier algorithm, and this research provides further evidence for the baseline viability of the random forest in the ship classification task.

Results show the highest accuracies were achieved with features extracted using the HOG and LBP algorithms. Additionally, we noticed consistent accuracy improvement in the SPM augmented VBOW. These results appear to be strong evidence that encoding spatial and structural information in the feature vectors enhances the performance of the random forest.

Simulation results show there is a great difference in accuracy between the VBOW representation of resized and original images, especially when the RAW and LOG TF were applied. Results strongly suggest that random forest yields higher accuracies from vessel images of the same size and orientation. In contrast to the VBOW-SVM results, the VBOW-random forest results consistently showed little influence of collection frequency factors and normalization, suggesting limited applicability of these weighting schemes.

The VBOW-random forest performance was most responsive to vocabulary size when the Binary TF factor was applied, with the largest vocabularies yielding the highest accuracies. This finding suggests that the random forest effectively characterizes images based on the presence or absence of a visual word in a class. Results suggest the only viable feature representation for original images considered in this study is the BINARY TF weighted VBOW, as it was the only feature representation that yielded accuracies greater than 80%. There is ample opportunity for following research to optimize random forest performance and configure it for military applications.

Recall random forest can natively process any combination of numerical, categorical and ordinal data. The algorithm is thus uniquely suited to classify images based off features extracted from different algorithms or different sensors. Two possible examples of heterogeneous feature vectors are those that are a combination of features extracted from images and image metadata, and feature vectors that are composed of features from local and global extractors.

In this study the VBOW was formed by K-means clustering of the top 80% strongest SURF descriptors. Alternative descriptors such as SIFT can be used to form the VBOW. Also, varying the descriptor strength threshold used in forming the VBOW may improve the performance of the random forest. Visual word generation may be improved through alternate clustering methods such as hierarchical clustering, Gaussian mixture models, and self-organizing maps [54].

LIST OF REFERENCES

[1] U. Kanjir, H. Greidanus, and K. Oštir, "Vessel detection and classification from spaceborne optical images: A literature survey," *Remote Sensing of Environment*, vol. 207, pp. 1–26, Mar. 2018. [Online]. doi: 10.1016/j.rse.2017.12.033

[2] K. Rainey, S. Parameswaran, J. Harguess, and J. Stastny, "Vessel classification in overhead satellite imagery using learned dictionaries," in *Proc. SPIE 8499*, 2012. [Online]. doi: 10.1117/12.928875

[3] S. Parameswaran and K. Rainey, "Vessel classification in overhead satellite imagery using weighted 'bag of visual words'," in *Proc. SPIE 9476*, 2015. [Online]. doi: 10.1117/12.2177779.

[4] T. Wood, "What is a random forest?," DeepAI, Sep. 10, 2020. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/random-forest

[5] M. Belgiu and L. Drăguţ, "Random forest in remote sensing: A review of applications and future directions," *ISPRS Journal of Photogrammetry and Remote Sens.*, vol. 114, pp. 24–31, Apr. 2016. [Online]. doi: 10.1016/j.isprsjprs.2016.01.011

[6] K. Rainey, "Building an operationally relevant dataset from satellite imagery," in *IGARSS 2019 - 2019 IEEE Int. Geosci. and Remote Sens. Symp.*, 2019. [Online]. doi: 10.1109/IGARSS.2019.8900177

[7] K. Rainey, J. Reeder, and A. Corelli, "Convolution neural networks for ship type recognition," in *Proc. SPIE 9844*, 2016. [Online]. doi: 10.1117/12.2229366

[8] L. Huang, W. Li, C. Chen, F. Zhang, and H. Lang, "Multiple features learning for ship classification in optical imagery," *Multimed Tools Appl*, vol. 77, no. 11, pp. 13363–13389, Jun. 2018. [Online]. doi: 10.1007/s11042-017-4952-y

[9] Q. Shi, W. Li, R. Tao, X. Sun, and L. Gao, "Ship classification based on multifeature ensemble with convolutional neural network," *Remote Sen.*, vol. 11, no. 4, Jan. 2019. [Online]. doi: 10.3390/rs11040419

[10] H. Zhong, X. Song, and L. Yang, "Vessel classification from space-based AIS data using Random Forest," in *2019 5th Int. Conf. on Big Data and Inform. Analytics*, 2019. [Online]. doi: 10.1109/BigDIA.2019.8802792

[11] B. Snapir, T. W. Waine, and L. Biermann, "Maritime vessel classification to monitor fisheries with SAR: Demonstration in the North Sea," *Remote Sens.*, vol. 11, no. 3, Jan. 2019. [Online]. doi: 10.3390/rs11030353

[12] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *2005 IEEE Comput. Soc. Conf. on Comput. Vision and Pattern Recognition (CVPR’05)*, 2005. [Online]. doi: 10.1109/CVPR.2005.177

[13] S. Mallick, “Histogram of oriented gradients,” Learn OpenCV, Dec. 6, 2016. [Online]. Available: https://www.learnopencv.com/histogram-of-oriented-gradients/

[14] MathWorks, “Extract histogram of oriented gradients (HOG) features,” Accessed Nov. 11, 2020. [Online]. Available: https://www.mathworks.com/help/vision/ref/extracthogfeatures.html

[15] T. Ahonen, A. Hadid, and M. Pietikainen, “Face description with local binary patterns: Application to face recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, Dec. 2006. [Online]. doi: 10.1109/TPAMI.2006.244

[16] L. Armi and S. Fekri-Ershad, “Texture image analysis and texture classification methods - a review,” *Int. Online J. Image Process. and Pattern Recognition*, vol. 2, no. 1, pp. 1–29, 2019. [Online]. Available: https://arxiv.org/abs/1904.06554

[17] A. Hadid, “The local binary pattern approach and its applications to face analysis,” in *2008 1st Workshops Image Process. Theory, Tools and Appl.*, 2008. [Online]. doi: 10.1109/IPTA.2008.4743795

[18] T. Ojala, M. Pietikainen, and D. Harwood, “Performance evaluation of texture measures with classification based on Kullback discrimination of distributions,” in *Proc. 12th Int. Conf. Pattern Recognition*, 1994. [Online]. doi: 10.1109/ICPR.1994.576366

[19] M. Pietikäinen, “Local binary patterns,” *Scholarpedia*, vol. 5, no. 3, p. 9775, Mar. 2010. [Online]. doi: 10.4249/scholarpedia.9775

[20] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002. [Online]. doi: 10.1109/TPAMI.2002.1017623

[21] T. Ojala, M. Pietikäinen, and T. Mäenpää, “Gray scale and rotation invariant texture classification with local binary patterns,” in *Comput. Vision - ECCV 2000*, 2000. [Online]. doi: 10.1007/3-540-45054-8_27

[22] I. L. Kambi Beli and C. Guo, “Enhancing face identification using local binary patterns and k-nearest neighbors,” *J. Imaging*, vol. 3, no. 3, Sep. 2017. [Online]. doi: 10.3390/jimaging3030037

[23] MathWorks, “Extract local binary pattern (LBP) features,” Accessed: Oct. 18, 2020. [Online]. Available: https://www.mathworks.com/help/vision/ref/extractlbpfeatures.html

[24] T. Lindeberg, “Scale Invariant Feature Transform,” *Scholarpedia*, vol. 7, no. 5, p. 10491, May 2012. [Online]. doi: 10.4249/scholarpedia.10491

[25] Q. Li and X. Wang, “Image classification based on SIFT and SVM,” in *2018 IEEE/ACIS 17th Int. Conf. Comput. and Inform. Sci. (ICIS)*, 2018. [Online]. doi: 10.1109/ICIS.2018.8466432

[26] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. doi: 10.1023/B:VISI.0000029664.99615.94.

[27] F.C. Huang, S.Y. Huang, J.W. Ker, and Y.C. Chen, “High-performance SIFT hardware accelerator for real-time image feature extraction,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 3, pp. 340–351, Mar. 2012. [Online]. doi: 10.1109/TCSVT.2011.2162760

[28] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” in *Comput. Vision – ECCV 2006*, 2006. [Online]. doi: 10.1016/j.cviu.2007.09.014

[29] G. Yang, D. Li, G. Ru, J. Cao, and W. Jin, “Body height estimation system based on binocular vision,” *Int. J. Online Eng. (iJOE)*, vol. 14, no. 4, p. 177, Apr. 2018. [Online]. doi: 10.3991/ijoe.v14i04.8400

[30] H. Shuo, W. Na, and S. Huajun, “Object tracking method based on SURF,” *AASRI Procedia*, vol. 3, pp. 351–356, 2012. [Online]. doi: 10.1016/j.aasri.2012.11.055

[31] J. Yang, Y. Jiang, A. G. Hauptmann, and C. Ngo, “Evaluating bag-of-visual-words representations in scene classification,” in *Proc. 9th ACM SIGMM Int. Workshop Multimedia Inform. Retrieval*, 2007. [Online]. doi: 10.1145/1290082.1290111

[32] J. C. Niebles and R. Krishna, “Visual bag of words,” Lecture for CS131: Computer Vision: Foundations and Applications, Dept. of Comp. Sci., Stanford University, Palo Alto, CA, USA, fall 2018. [Online]. Available: http://vision.stanford.edu/teaching/cs131_fall1819/files/14_BoW_bayes.pdf

[33] P. X. Nguyen and L. Q. Hieu, “A new improved term weighting scheme for text categorization,” in *Knowledge and Systems Engineering*, vol. 244. V. Huynh, T. Denoeux, D. Tran, A. Le, and S. Pham, Eds. Cham, Switzerland: Springer, 2014, pp. 261–270. [Online]. doi: 10.1007/978-3-319-02741-8_23

[34] Y. Ko, “A study of term weighting schemes using class information for text classification,” in *Proc. 35th Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval*, 2012. [Online]. doi: 10.1145/2348283.2348453

[35] M. Fargues, “Support vector machine (SVM) & support vector regression (SVR),” Lecture for EC4440: Statistical Digital Signal Processing, Dept. of Elect. and Comput. Eng., Naval Postgraduate School, Monterey, CA, USA, fall 2019.

[36] S. Karamizadeh, S. M. Abdullah, M. Halimi, J. Shayan, and M. J. Rajabi, “Advantage and drawback of support vector machine functionality,” in *2014 Int. Conf. Comput., Commun., and Control Technol. (I4CT)*, 2014. [Online]. doi: 10.1109/I4CT.2014.6914146

[37] J. Kindermann, E. Leopold, and G. Paass, “Multi-class classification with error correcting codes,” in *Principles of Data Mining and Knowledge Discovery*, *5th Eur. Conf., PKKD 2001*, 2001. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.31.6566&rep=rep1&type=pdf

[38] MathWorks, “Support Vector Machines for binary classification,” Accessed Nov. 14, 2020. [Online]. Available: https://www.mathworks.com/help/stats/support-vector-machines-for-binary-classification.html#bsr5b42

[39] MathWorks, “Classify observations using support vector machine (SVM) classifier,” Accessed Nov. 14, 2020. [Online]. Available: https://www.mathworks.com/help/stats/classreg.learning.classif.compactclassificationsvm.predict.html

[40] MathWorks, “Fit multiclass models for support vector machines or other classifiers,” Accessed Nov. 14, 2020. [Online]. Available: https://www.mathworks.com/help/stats/fitcecoc.html#bufm0zb-1

[41] M. Galarnyk, “Understanding decision trees for classification (Python),” Medium, Oct. 18, 2020. [Online]. Available: https://towardsdatascience.com/understanding-decision-trees-for-classification-python-9663d683c952

[42] “Recursive partitioning,” class notes for STAT555: Statistical Analysis of Genomics Data, Eberly College of Sci., The Pennsylvania State University, University Park, PA, USA, 2018. [Online]. Available: https://online.stat.psu.edu/stat555/node/100/

[43] T. C. Au, “Random forests, decision trees, and categorical predictors: The ‘absent levels’ problem,” J. Mach. Learning Res., vol. 19, no. 45, pp. 1 - 30, Sep. 2018. [Online]. Available: https://www.jmlr.org/papers/volume19/16-474/16-474.pdf

[44] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997. [Online]. Available: https://www.cs.princeton.edu/courses/archive/spr07/cos424/papers/mitchell-dectrees.pdf

[45] L. Breiman, “Random forests,” *Mach. Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001. [Online]. doi: 10.1023/A:1010933404324

[46] L. Breiman, J. Friedman, R. Olshen, and C. Jones, *Classification and Regression Tree*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984. [Online]. doi: 10.1201/9781315139470

[47] V. Zhou, “A simple explanation of Gini impurity,” Victor Zhou, Mar. 29, 2019. [Online]. Available: https://victorzhou.com/blog/gini-impurity/

[48] W.Y. Loh, “Classification and regression trees,” *Wiley Interdiscip. Rev.: Data Mining Knowl. Discov.*, vol. 1, no. 1, pp. 14–23, Jan. 2011. [Online]. doi: 10.1002/widm.8

[49] L. Breiman and A. Cutler, “How random forests work,” Random Forests. Accessed Oct. 27, 2020. [Online]. Available: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr

[50] S. Raschka, “Model evaluation 2: Confidence intervals,” Lecture for STAT479: Machine Learning, Dept. of Statist., University of Wisconsin-Madison, Madison, WI, USA, fall 2018. [Online]. Available: https://sebastianraschka.com/pdf/lecture-notes/stat479fs18/09_eval-ci_slides.pdf

[51] L. Breiman, “Notes on setting up, using, and understanding fandom forest V3.0.” Accessed: Nov. 13, 2020. [Online]. Available: https://www.stat.berkeley.edu/~breiman/Using_random_forests_v3.00.pdf

[52] R. Cristi, “SIFT and SURF implementation,” Lecture for EC4480: Image Processing and Recognition, Dept. of Elect. and Comput. Eng., Naval Postgraduate School, Monterey, CA, USA, fall 2016.

[53] MathWorks, “Error (misclassification probability or MSE),” Accessed: Oct. 31, 2020. [Online]. Available: https://www.mathworks.com/help/stats/treebagger.error.html

[54] MathWorks, “Cluster analysis.” Accessed: Nov. 18, 2020. [Online]. Available: https://www.mathworks.com/discovery/cluster-analysis.html

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California