UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Automated Detection of Mine-Like Objects in Side Scan Sonar Imagery /

Permalink

https://escholarship.org/uc/item/4gw8g426

Author

Barngrover, Christopher M.

Publication Date

2014

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Automated Detection of Mine-Like Objects in Side Scan Sonar Imagery

A dissertation submitted in partial satisfaction of the
requirements for the degree of Doctor of Philosophy

in

Computer Science

by

Christopher M. Barngrover

Committee in charge:

Ryan Kastner, Chair
Serge Belongie
Jules Jaffe
Truong Quang Nguyen
Lawrence Saul

2014

Copyright

Christopher M. Barngrover, 2014

All rights reserved.

The Dissertation of Christopher M. Barngrover is approved and is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2014

DEDICATION

I would like to dedicate this research to all the people in my life who supported me through the process. Specifically my parents, Timothy Barngrover and Joan Barngrover, who have been a constant source of reassurance and motivation throughout my entire academic career, culminating with this research. My brother, Benjamin Barngrover, for pushing me to achieve at the highest level I can. My adviser since year two, Ryan Kastner, who I consider a mentor but also a friend. And finally to my fiancé, Kara Savena, who has supported through the most demanding and stressful portion of the process. Without her help and encouragement, I would not have made such progress while working full time over the final nine months.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to acknowledge Professor Ryan Kastner for his support from the early stages of my research until the very end and as the chair of my committee. We went through many iterations of many papers with a few setbacks and without his guidance this would not have been possible.

I would also like to acknowledge the feedback I received from Professor Serge Belongie along the way. His expertise in the field of computer vision was essential to my research and my progress.

The support of Gerry Hong, Michael Stuckenschneider, and Daniel Kichura from the Space and Naval Warefare Systems Center Pacific was invaluable. This team helped with the data collection and gave advice from a users point of view.

Finally, I would like to acknowledge Brett Ryan, Alric Althoff, and Paul DeGuzman for their important collaborations as co-authors and to J. Tory Cobb of Naval Surface Warfare Center Panama City for his support and guidance on parts of this research.

Chapter 4, in full, is a reprint of the material as it appears in IEEE Journal of Oceanic Engineering, Barngrover, Chris; Belongie, Serge; Kastner, Ryan, 2014. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in full, has been submitted for publication of the material as it may appear in Ocean Engineering, Barngrover, Chris; Ryan, Brett, Belongie, Serge; Kastner, Ryan, 2014. The dissertation author was the primary investigator and author of this paper.

Chapter 8, in full, has been submitted for publication of the material as it may appear in IEEE Journal of Oceanic Engineering, Barngrover, Chris; Althoff, Alric; DeGuzman, Paul; Kastner, Ryan, 2014. The dissertation author was the primary investigator and author of this paper.

VITA

2005	Bachelor of Science, Computer Science Bachelor of Science, Mathematics Purdue University
2005	Scientist, Unmanned Systems Branch Space and Naval Warfare Systems Center Pacific
2010	Master of Science, Computer Science University of California, San Diego
2014	Doctor of Philosophy, Computer Science University of California, San Diego

PUBLICATIONS

"JBoost Optimization of Object Detectors for Autonomous Underwater Vehicle Navigation" International Conference on Computer Analysis of Images and Patterns (CAIP), August 2011.

"The Stingray AUV: A Small and Cost-Effective Solution for Ecological Monitoring", IEEE Oceans, September 2011.

"Semisynthetic Versus Real-World Sonar Training Data for the Classification of Mine-Like Objects", Oceanic Engineering, IEEE Journal of, 2014, vol. PP, no. 99, pp. 1-9.

"Multi-Feature Selection Framework for the Detection of Mine-Like Objects in Side Scan Sonar Imagery", Ocean Engineering, 2014, submitted.

"A Brain-Computer Interface (BCI) for the Detection of Mine-Like Objects in Side Scan Sonar Imagery", Oceanic Engineering, IEEE Journal of, 2014, submitted.

ABSTRACT OF THE DISSERTATION

Automated Detection of Mine-Like Objects in Side Scan Sonar Imagery

by

Christopher M. Barngrover

Doctor of Philosophy in Computer Science

University of California, San Diego, 2014

Professor Ryan Kastner, Chair

The task of detecting mine-like objects (MLOs) in side scan sonar imagery has a profound impact on military operations. The current process involves subject matter experts analyzing sonar images searching for MLOs. The automation of this problem has been heavily researched over the years without a definitive solution that outperforms the manual approach in real world scenarios. This paper presents a series of approaches and experiments centered on the use of GentleBoost feature selection classifiers for the detection of MLOs in side scan sonar. In a comparison of semi-synthetic versus real world training data with two different boosted single-feature selection classifiers, we see

that semi-synthetic data can provide insight in to potential performance of a classifier. We run experiments training and testing GentleBoost single-feature classifiers on six different feature types, finding that the Haar-like feature classifier performs the best. We propose a GentleBoost multi-feature selection framework that allows for multiple feature types to be in the pool of selectable features, finding that a combination of Haar-like features, speeded up robust features (SURF), and simple shadow features performs better than the Haar-like feature classifier. Experiments with tiered, or cascaded, classifiers show a reduction in false positives for lower true positive rates. The multiple instance learning (MIL) approach shows great potential for future efforts, achieving improved true positive rates at higher false positive rates. A final approach considers the complimentary benefits of computer vision and human vision, introducing two brain-computer interface (BCI) systems. One BCI uses the Haar-like feature classifier as a first stage cascaded in to a human processing second stage. The other adds a third stage that employs a novel support vector machine (SVM) classifier based on the Haar-like feature and human interest scores from multiple subjects. Overall, our GentleBoost feature selection classifier variations result in performance improvement for the detection of MLOs in side scan sonar imagery.

Chapter 1

Introduction

Automated detection of targets such as mine-like objects (MLOs) in underwater environments is a difficult task due to the inherent complexity of capturing images. The sonar is a prominent sensor for manual and automated detection of objects due to its ability to visualize these dynamic environments. The underwater acoustics produce images with highlights and shadows for objects protruding from the sea floor, even in low visibility environments. The side scan sonar, specifically is excellent at mapping the sea floor.

For the specific task of detecting MLOs, the current process employed involves collecting large amounts of side scan sonar data and having trained operators search the images for targets of interest, which is very time consuming. Automating this task would be of great value, allowing for clearing larger areas in short time periods, saving cost and potentially lives.

There is a vast amount of research on techniques for automatically detecting MLOs in side scan sonar. The earlier research relied primarily on models of the targets, focusing on the shadow and highlight characteristics [17, 38, 49, 33]. Some research uses machine learning such as neural networks [67, 33] with more recent efforts using computer vision local features and boosting [54].

This paper focuses on the use of boosting as a training capability for various

features, considering the model style features as well as the local descriptors. For part of the research we simply use a basic GentleBoost single-feature selection algorithm similar to other research [54]. Then we consider many variations of the boosted feature selection process and many different features. We also consider a brain-computer interface (BCI) that employs a boosted single-feature classifier and a human interest classifier using electroencephalography (EEG).

There are a number of major contributions of this research, all related to the detection of MLOs in side scan sonar. We list these contributions here in the order they are presented in the following chapters rather than in order of impact:

- A survey of the work done on this topic. This is the foundation of a survey paper to be published in collaboration with other experts in the field.
- An evaluation of the use of semi-synthetic datasets for classifier generation in lieu of real world datasets, specifically for classifying MLOs in side scan sonar, and quantifying the relative improvement when training on real world data instead of semi-synthetic.
- Access to the semi-synthetic and real world datasets for the academic community. The real world training and testing datasets are of particular significance.[1]
- Evaluation of prominent and unique computer vision features with the GentleBoost feature selection algorithm.
- The introduction of the multi-feature selection framework, which is structured such that any dataset and any group of features can be used for training and classification. The concept involves allowing the boosted selection algorithm to consider more than one type of feature during training.

[1] http://kastner.ucsd.edu/datasets/barngrover/

- The proposition of applying a multi-feature selection process for the detection of mine-like objects in side scan sonar imagery.

- The optimal combination of feature types selected by the multi-feature selection framework, which uses the Haar-like feature, the speeded up robust feature and a simple shadow feature. The resulting *HAAR + SURF+ SHADOW* classifier improves the true positive rate (TPR) by 12.69% up to 88.56% at approximately 3.0 false positives per square kilometer (FP per km^2) and under other parameters it increases TPR by 3.79% up to 94.56%, while reducing FP per km^2 by 1.74.

- The introduction of two brain-computer interface (BCI) systems. One BCI uses the Haar-like feature classifier as a first stage cascaded in to a human interest classifier second stage. The other adds a third stage that employs a novel support vector machine (SVM) classifier based on the Haar-like feature and human *interest scores* from multiple subjects.

The remainder of this paper is organized as follows. In Chapter 2 we explain in depth how sonars function and the process that creates the sonar image. This understanding of how the sonar image is created leads to better decisions when selecting features and a more complete knowledge base for the research. Then in Chapter 3 we present the survey on the state of research on the topic of automated MLO detection. A comparison of single-feature boosted classifiers trained on semi-synthetic versus real data is considered in Chapter 4. Next in Chapter 5 we present the primary contributions of this paper. This chapter introduces a multi-feature selection framework and an optimized classifier trained by this algorithm. The concept of tiered classifier processing is proposed and evaluated in Chapter 6. Multiple instance learning has been shown to work well for many scenarios such as tracking objects. In Chapter 7 we propose applying this concept to the detection of MLOs using the Haar-like feature classifier. Another major contribution

comes form Chapter 8, which considers different BCI systems using computer vision and human vision classifiers together for the task of MLO detection. Finally, Chapter 9 concludes the paper and summarizes its contents concisely.

Chapter 2

Side Scan Sonar Imagery

In order to understand this paper and all of the various research paths related to the overall goal of detection and classification of mine-like objects in side scan sonar, it is imperative to understand the fundamentals of sonars. This chapter will touch on the history of sonar and explain some of the properties of acoustic imaging.

2.1 History

Sound was first used to examine the seabed by A B Wood and some colleagues with the development of the prototype echosounder in 1929 [66]. Following this break through there were many variations of the prototype considered. Most of the echosounders of the time utilized bottom penetrating low frequency sound sources [21]. With the antisubmarine warfare effort of World War II came a large surge in research into acoustic applications for mapping the sea floor. This research formed the technical base for the modern side scan sonar [57]. It was not until 1958 that the back-scatter from high frequency sound waves were used to create maps of geological features on the sea floor [8]. In 1961 the first functional side scan sonar was developed and installed on the RRS Discovery II research vessel. This sonar operated at a frequency of 36kHz, which is optimal for large scale survey such as for continental shelf science, and requires a swath width of 1500 m in water depths up to 200 m [61]. In the same year the first

major scientific paper focusing on the use of side scan sonar to survey the sea floor was published [18].

During these early stages of the side scan sonar the hardware was referred to by a variety of names. Eventually the British began using the acronym ASDIC to refer to these systems. The exact meaning of this acronym is debatable, but the official claim of the British Admiralty is that ASDIC stands for Allied Submarine Detection Investigation Committee, though no committee with this name has been found [26]. Meanwhile, the American scientists had begun referring to the systems as SONAR for SOund Navigation And Ranging [59]. Obviously the term sonar has become the more widespread label for such systems and is no longer capitalized as an acronym.

2.2 Underwater Acoustics

In order to fully understand how active sonars produce usable imagery of the sea floor it is important to have a grasp on the fundamental concepts of acoustics in the underwater environment. This includes the broad information about the speed of sound and the relationships between the parameters of the sonar and the output image, as well as the important highlight and shadow phenomenon. In addition, the propagation of sound through the water has many distortions including absorption and refraction that greatly affect the result of active sonars.

2.2.1 Acoustic Concepts

The underlying concepts of sound are a logical place to start for a strong foundation in understanding sonar. Sound is the mechanical vibration which is produced by longitudinal waves through water or any other elastic medium. Acoustics are useful for underwater applications because sound waves travel through water very efficiently, especially compared to light or radio waves [3].

The speed of sound as it propagates through the water is an essential variable when creating a sonar, because it allows for the estimation of target range based on the time between echo transmission and return. However, it is not necessary for understanding the resulting images. The main thing to note is that the velocity of sound through water is different from the velocity in air and that the actual speed is determined by temperature, salinity, and pressure from depth. The typical range of velocity in most fresh and seawater conditions is from 1,405 to 1,550 meters per second [3].

The frequency is represented by the variable f and it is the number of waves that pass through one location over the period of one second. A Hertz (Hz) is the simplest measurement of frequency, where 1Hz is one wave in one second. Sonars are mostly represented in kilohertz (kHz) or 1,000 Hz and occasionally in megahertz (MHz) or 1,000,000 Hz. The wavelength is represented by the variable λ and is the distance between each individual waveform. The frequency and wavelength of the transmitted sound wave by a sonar affects the output image in specific ways, which will be further discussed in Section 2.3.

The sonar equation is an essential part of understanding the sonar and the output image. It shows the relationship between all elements of the sound wave traveling from a transducer to a target and returning, as well as any unexpected movements of the sound after transmission. It is presented here in Equation 2.1 in terms of how much energy must return to the transducer in order to be detected.

$$DT \leq SL - 2 \times TL + TS - (NL - DI) \tag{2.1}$$

The detection threshold (DT) represents the sensitivity of the sonar system to detect acoustic energy. The acoustic response must be greater than this threshold to register in the system. The source level (SL) is the strength of the initial acoustic energy as

introduced by the sonar. As the sound waves travel through the water there is transmission loss (TL) caused by both distortions and the attenuation of water. This is multiplied by two in order to account for the travel to and from the target. The target strength (TS) represents the reflectivity of any target with which the sound wave collides. This value is determined by a combination of size, shape and composition. The noise level (NL) in the water column is the amount of background sound that might disrupt the sonar's sound waves. Finally, the directivity index (DI) represents the narrowness of the beam of sound produced by the sonar. It is subtracted out from the noise level, which represents the noise level throughout the water column and not just in the narrow band of sound given by the directivity index.

The details of these fundamental acoustic concepts help to appreciate the math behind a sonar system and how many elements affect the output image. Having a high level understanding of these concepts is useful when analyzing sonar images. The concepts also lay a foundation for understanding some of the important characteristics of sonar images, which exemplify all targets of interest.

2.2.2 Highlights and Shadows

In the images created by side scan sonars, the pair of a highlight region followed laterally by a shadow region is common to all targets of interest. There are many factors that will determine the relative brightness of the highlight and the size of the shadow. Understanding these factors is key to determining if a highlight and shadow pair represents a target with a certain size and shape.

With all other variables of the sonar equation being equal, the target strength (TS) is based on the acoustic reflectivity of the given target. It is important to note that this is not limited to targets on the seabed but includes mid-water and surface targets with enough reflectivity to be detected by the sonar. The reflectivity of a target

is based on a complex relationship between the target's composition, size, thickness, shape, and roughness and the sonar's frequency, pulse duration, and incident angle of the waveform [3].

The target composition is a fairly straight forward factor, where the acoustically dense materials are more reflective than acoustically soft. The acoustic hardness of an object depends on the relative speed of sound through the object compared to the elastic medium, such as water. The thickness of the target is important here, because the change in speed of sound must continue for multiple wavelengths [3].

The size of the target and the incident angle of the acoustic beam must be such that the entire beam hits the surface. If a target is smaller than the beam width or if the incident angle is such that the entire beam does not hit the target surface, there will not be a strong reflection. The shape also plays a factor because it determines the direction of reflection. For instance a circular surface will reflect the waveforms in many directions.

Unlike the reflectivity, the acoustic shadow is not as dependent on variables of the target's surface and composition. The shadow is created because the waveforms hit some other target and are blocked from the seabed directly behind the target. The result is the lack of any data return from the area where the waveforms cannot reach. This is shown visually in the first image of Figure 2.1 where the small peak blocks the waveforms from hitting seabed.

The shadow has the powerful capability to show information about the vertical shape and features of a target proud of the seabed. The lateral distance of the shadow is based on the range to target, vehicle altitude, and height of the target. Sometimes detailed features about an object can be seen in the acoustic shadow, as with the bicycle shown in the second image of Figure 2.1.

Acoustic shadows are also created by bottom depressions, where the depression depth is low enough to cause the standard seabed to block acoustic waveforms from

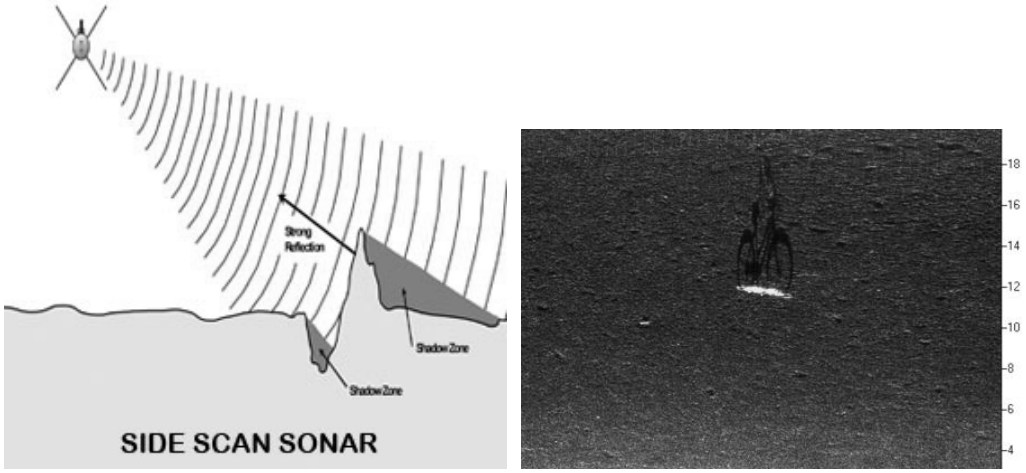


Figure 2.1. The left image provides a visual explanation of how an acoustic shadow is produced. The top of the depression and the peak block the acoustic waves from reaching the seabed behind. The right image shows how useful the shadow can be in practice, providing a high resolution view of the object's shape features.

reaching the depression. The altitude of the vehicle and the size of the depression will determine the shadow. The important distinction with these types of shadow is that there is no reflective target to create a highlight before the shadow. This is why the combination is so important when searching for targets of interest.

2.2.3 Geometric Distortions

The elastic medium through which the sound waveforms travel has various characteristics leading to the transmission loss (TL) in the sonar equation presented as Equation 2.1. The major distortions of the waveforms in water are absorption, refraction and multi path.

Acoustic absorption occurs when the sound wave travels through the water and the acoustic energy causes water molecules to collide thus converting into heat. The absorption coefficient represents the amount of absorption that will occur under the given parameters of frequency, salinity, temperature and pressure [3].

Acoustic refraction is when the sound wave travels through different water conditions causing the sound to refract. This refraction, or bending, always occurs towards the water configuration yielding slower velocity in accordance with Snell's Law [3]. Thermoclines are layers of temperature and haloclines are layers of salinity, both of which can cause refraction, along with layers of pressure. Acoustic refraction presents itself as blurred and wave-like patterns in the output image.

When the beam of acoustic waves hit a reflective surface, such as the seabed or the water surface, it reflects the waves in many directions. Until now we have focused on the waveforms returned to the sonar system, but the others are important as well. Acoustic multipath refers to the waves that are reflected in other directions, but eventually reflect again and return to the sonar system [3]. The sonar plots these returns based on time and reflective targets with multipath appear as duplicates in the output image.

There is another important geometric distortion that is inherent to the movement of the vehicle housing the sonar system. Any variation in trajectory, speed or orientation of the vehicle can cause distortions in the output image. Wind and current, as well as controls or sensor imperfections can cause instabilities in vehicle movement. The translation to the output image might be errors in aspect-ratio or other noticeable geometric distortions [10].

All of these geometric distortions emphasize the difficulty in sonar imaging, which has an abundance of variables involved in the process. Understanding the possible distortions, including the causes and the results, improves the ability to process sonar images when searching for specific targets of interest.

2.3 Sensor Overview

There are many different types of sonar systems with varying parameters for frequency, wavelength, beam angle, pulse length, bandwidth, etc. This section outlines some of the tradeoffs on these parameters with a focus on the specific type of sonar

system utilized in this research, known as side scan sonar.

2.3.1 Configuration

The side scan sonar is attached to each side of the autonomous underwater vehicle (AUV) or a tow fish, which is pulled behind a boat or submarine. Each sensor emits an acoustic wave over regular intervals at a downward angle and records the reflected wave. Each transmission is associated with returns to create one line of the output image. This concept is explained visually in Figure 2.2. This type of sonar image has a blind spot under the vehicle where the two sonars do not focus their beams in order to avoid crosstalk [40].

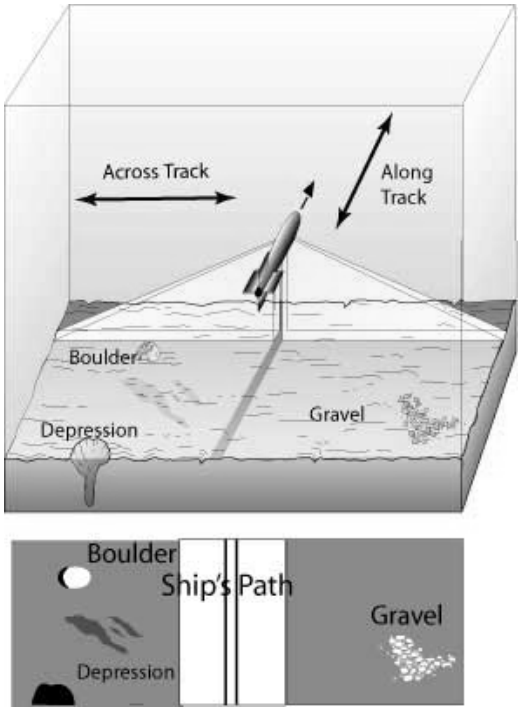


Figure 2.2. Image courtesy of Woods Hole Science Center. The figure shows the image creation from side scan sonar. The triangles under the vehicle show the acoustic beams. The area behind the vehicle has already been captured in the image representation at the bottom, with important features labeled.

2.3.2 Coverage and Resolution

The coverage, also known as range, and the resolution from a sonar system have an inverse relationship. Altering certain parameters may increase resolution but at a reduced coverage, while a lower resolution allows for more coverage. The higher resolution simply takes more time to achieve the same coverage.

The range for a side scan sonar image refers to the lateral coverage distance in meters from each side of the vehicle. For a given vehicle with a set sonar system, the resolution of the output image can be increased by lowering the altitude of the vehicle, but this decreases the coverage. Figure 2.3 shows in rectangles two sonar images of the same type of mine object at 30 meter range on the left and 10 meter range on the right. Notice that the mine appears larger and more detailed in the 10 meter range image.

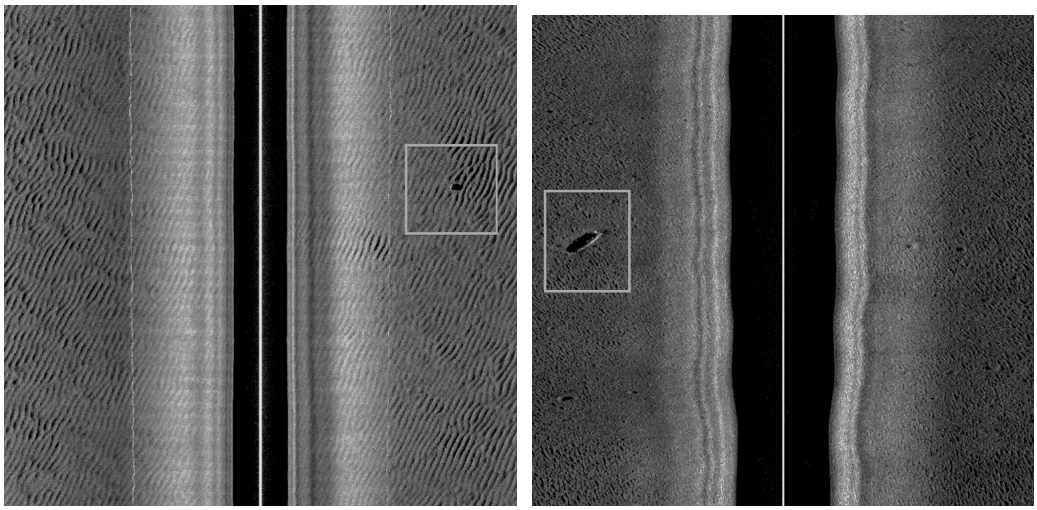


Figure 2.3. Side scan sonar at 30 meter range on the left compared to 10 meter range on the right. The rectangular regions show similar mines in each type of range data.

Beam angle is one of the parameters that is tied to the coverage and resolution dichotomy. The beam angle is the width of the focused acoustic transmission in degrees. The smaller the beam angle, the more concentrated the engery and the higher the resolution at a cost of reduced coverage. Similarly, a lower frequency can travel further

allowing for better coverage, but higher frequencies provide better resolution [3].

There is also a trade off between the bandwidth and pulse length of the sonar system. Longer pulse lengths increase range, while shorter pulse lengths increase resolution. However, the allowable pulse length is dependent on the receiver bandwidth of the system and a given system has an optimal pulse length for the chosen receiver bandwidth [3].

The resolution of an image is important when the goal is identification of targets of interest. The resolution must be high enough to detect the target in question in terms of its size. The transverse resolution, or along-track resolution, is the resolution on the y-axis of the output image. In order to differentiate between two targets, the beam angle must be less than the distance between the objects. Range resolution, or cross-track resolution, is the resolution on the x-axis of the output image. The pulse length and frequency together determine the ability to differentiate between two targets [3].

2.3.3 Time Varied Gain

The sonar equation shows us that there is transmission loss (TL) over the distance of travel by the acoustic waveform. As a result, new sonar technology attempts to account for this loss of signal. The concept of Time Varied Gain (TVG) provides increasing amplification to received signal over time. In other words, the returns from further down range are amplified to look more consistent across the image.

Using the wrong TVG curve in a system can cause the output image to look washed out. Most new sonar systems have the capability to choose the correct TVG curve and only see this error when the TVG is operated in manual mode.

2.4 Marine Sonic Sonar Image

The application nature of this research requires the capability of processing the proprietary sonar image format directly. For this research the sonar is made by Marine Sonic and their sonar image format is called Marine Sonic Tagged Image File Format (MSTIFF). This section explains this specific file format as well as describes an application for viewing the sonar images and clicking to see geodetic and altitude information pulled from the file.

2.4.1 MSTIFF

The Marine Sonic Tagged Image File Format (MSTIFF) basically acts as a wrapper around the commonly known Tagged Image File Format (TIFF) allowing for additional information common to sonar images.

The MSTIFF file begins with an 8-byte image header identifying it as an MSTIFF and pointing to the actual data. The actual data includes sonar and navigational data. For our research we care about the bitmap parts of the sonar data and certain fields of the navigational data.

The sonar data is split into the right and left channels since the data comes from two different sonars for either side of the vehicle. Utilizing some of the fields to determine size and direction the data was collected the image can be reconstructed in grayscale or in a bronze color format.

The navigational data is a bit more complicated to convert to a useable format. We developed functions to convert a given location of the image to the desired navigational related information. Specifically we pull the altitude of the vehicle, which is connected to a given y-location in the image associated with a certain ping. Each individual location also provides a latitude and longitude. Finally we also capture the course over ground

(COG) direction of the vehicle.

2.4.2 Show Image Application

The Show Image Application has two uses in this research. Originally it was a way to test the library that handled the MSTIFF sonar image file format for accuracy. But the secondary use proved more important, as the application became essential for easily identifying the location of mines in new image datasets based on ground truth.

The Show Image Application processes a directory which can be navigated with the buttons at the bottom. Given the ground truth locations of mines in the test field, the application will highlight ground truth locations in any image. This is done by changing any pixel with a matching latitude and longitude to bright green. This is the approximate location of ground truth and often the actual mine will be near by in the same sonar image or in a following image.

Figure 2.4 shows an example of processing a MSTIFF sonar image. The gray rectangular region on the left of the image shows the ground truth and the dark box just above and to the left is around the labeled mine. Since this image had already been labeled, the box is pulled from the label file. The figure also shows all of the navigational information for the location of the mine based on a mouse click.

The library used to process each MSTIFF file in the Show Image Application is shared with our primary research code. This way, we can process the actual sonar image as we would in real time on a vehicle. This processing is currently slower than a direct TIFF image processing, allowing room for optimization. It is still very fast and allows for the additional use of navigational information to aid in the object detection task.

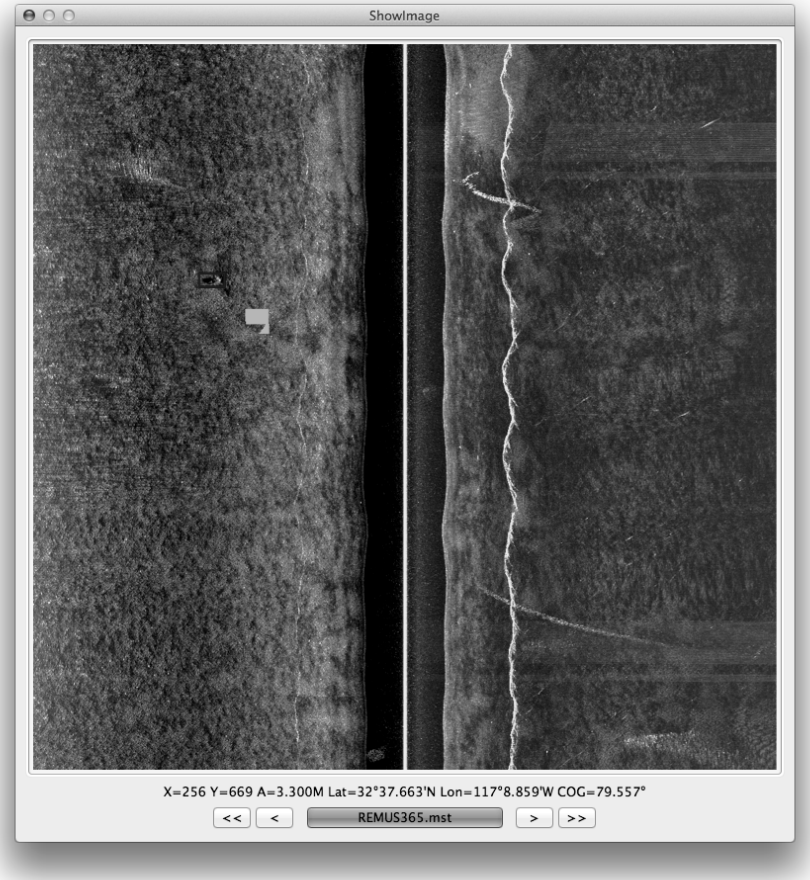


Figure 2.4. The Show Image Application shows the location of pixels near known ground truth as green, which in this figure are the gray rectangular shape on the left side of the image. The application also shows a box around any locations labeled as positive in our label file, Which is a dark rectangle around the mine to the upper left of the gray pixel region. Finally the application provides vehicle information at the time of sonar image capture for a clicked point on the image. Course over ground of the vehicle is shown as COG.

Chapter 3

A Survey of Object Detection in Side Scan Sonar Imagery

Side scan sonar images are not well defined due to speckle noise and the nature of the environment causing spurious shadows, sidelobe effects, and multipath returns [19]. The result is large variability in targets, clutter, and background signatures, which causes the detection of mine-like targets and the classification of the mines to be a complicated problem.

However, most of the literature agrees on the basic signature of a mine-like object in side scan sonar. The principles of the sonar and the properties of sound wave propagation underwater means that a region of interest will have distinct segments. Since a mine is usually made of denser material than other seabed objects, there will be a highlight segment corresponding to the mine. The mine will create a shadow segment orthogonal to the sonar by protruding from the sea floor and blocking the sound waves. There will also be segments of clutter surrounding the highlight and shadow segments. Finally the region of interest will have a background segment corresponding to the reverberation from the sea floor [48].

3.1 Image Enhancement

Throughout the literature the first step for any object detection and classification algorithm in side scan sonar is some sort of image enhancement [40, 60, 64, 65, 9, 33, 32]. The preprocessing may use one or many of a variety of techniques, but the end goal is always a normalized image with reduced noise. A normalized image means that throughout the image the background has certain pixel characteristics such that objects, both highlights and shadows, are consistently disparate from the background.

One type of image normalization is histogram equalization, where the image is transformed so that its histogram is nearly flat. A flat histogram means that each intensity band has nearly the same number of pixels from the image [40, 64]. Similar to using histograms, some algorithms attempt to equalize the intensity levels associated with different bottom environments through proprietary techniques [9, 32]. Another algorithm uses a serpentine forward-backward filter, which considers both sides of each pixel in order to normalize the individual pixel [60]. Regardless of the technique employed, the result is an image with consistent background, highlight, and shadow pixel intensities.

In addition to the normalization, noise reduction is important to the image enhancement step. A diamond or square shaped median filter, which averages a pixel based on certain neighbor pixels, will smooth the image to reduce noise [65]. Similarly, the Wiener filter considers local mean and variance for a small window to reduce single pixel intensity fluctuation [64]. Another technique uses wavelet based de-noising via Donoho's shrinkage, which involves choosing and shrinking a specific level in the wavelet. The Gaussian filter and Difference of Gaussians (DOG) could also be used to reduce noise and smooth the image [40].

The goal of the image enhancement step is to prepare the image in order to maximize the capability of the algorithm chosen for object detection and classification.

Therefore, certain image enhancement techniques will work best with certain detection and classification schemes.

3.2 Segmentation

Segmentation is the most common way to isolate the highlight and shadow regions associated with mines in sonar imagery [40, 60, 64, 65, 49, 38, 9, 33, 56, 41, 48]. The reason is the nature of sonar image creation, which causes the pixels representing the mine itself to be at a higher than average intensity, while the shadow created from the mine has a very low intensity.

The simplest way to segment the image is to establish threshold ranges in pixel intensity to represent the desired segments of background, highlight, and shadow [40, 65, 56, 33]. An improvement over the static thresholding is some form of adaptive thresholding, such as adjusting the threshold based on local mean [60] or based on local histograms [9, 48].

There are also more complicated techniques for segmenting an image, such as fuzzy functions [64, 33, 41] or using a Markov random field (MRF) [49, 38]. Some fuzzy functions include the fuzzy k-means clustering technique, which works on luminance mean and variance within small windows, or c-means clustering, which adds the use of grey level information. In general, fuzzy clustering is very sensitive to speckle noise [64]. Alternatively, the MRF model provides a reliable framework for segmentation since it utilizes pixel dependencies. Specifically the model takes into consideration the labels of the surrounding pixels, making a pixel surrounded by shadow pixels more likely to be labeled a shadow [49].

The result of any type of segmentation is an image where each pixel is labeled as one of a limited number of segment options. For example, a binary image will have object pixels labeled one and background pixels labeled zero. This simplifies the image

for the steps of the algorithm that follow.

3.3 Detecting Regions of Interest

Sometimes the segmentation leads to defining regions of interest simply by considering all segments that include highlight, shadow, or the combination [33, 49, 41]. Other times there are additional techniques used to detect regions of interest in the segmented or raw images. Whether directly following the segmentation or with an intermediate step, the result is a region of interest containing a potential mine-like object.

Many of the detection solutions involve some sort of matched or template filter [40, 9, 56, 48, 65]. The simple templates look for a highlight and shadow combinations in the image [40, 9, 65]. Some algorithms take the template a step further, including not just highlight and shadow regions, but also highlight clutter and shadow clutter regions [48].

A more complicated example may convolve the template with potential regions in the image and apply a threshold to determine if the region is a candidate. In this case the template needs to be general so that it covers all possible mine-like objects, but still specific enough to ignore most of the bottom clutter [56]. Another template matching technique creates a subspace of shapes with six vertices that represent each example mine-like object and then compares a considered object's normalized shape with the subspace of example shapes [50].

After these techniques have been applied to an image the algorithm will have some number of regions of interest to consider further. The goal is to reduce the number of regions that must be processed with expensive classification algorithms without missing any actual target objects.

3.4 Classifying Regions of Interest

The classification of a region of interest is generally a costly set of algorithms to determine if there exists an object of a certain class in the region. For the common case of mine-like objects, the classification will be binary: mine or not mine. The classification does not necessarily have to be binary, but the processing becomes more complicated with the increase of classes.

In order to classify the region of interest, a set of features must first be extracted. Sometimes this is a simple calculation of distance [50], signal-to-noise ratio and shape [9], or area, pixel amplitude and parallel lines via the Hough Transform [41]. Other times there is a dedicated feature extraction algorithm such as a wavelet packet-based feature extraction scheme [67] or canonical correlation analysis (CCA) to extract features, which can simultaneously segment the image while extracting the features [60].

There has been much research in classification techniques, causing this step to have the most variety in the literature. The easiest classification technique is to threshold the extracted features based on example data [9, 41]. Some take this a step further by using the log-likelihood ratio of the features in order to classify the region of interest [50, 9, 60].

Some algorithms use the extracted features as training input into a neural network such as a two-layer back propagation neural network (BPNN) [67] or a probabilistic neural network [33, 34]. These neural networks use the training data to learn the range of values for each feature type that best characterize the target. These values are then used as indicators to classify a candidate region of interest.

The K-Nearest Neighbor (K-NN) is a technique where the k nearest neighbor regions, which are defined by training data, are used as evidence when classifying the candidate region. The distance of the neighbor to the candidate region affects the belief

score, with smaller distances causing higher belief scores [34].

A cooperating statistical snake (CSS) model can recognize the boundaries of highlights and shadows in the image. Specifically, two statistical-snakes are used to find the object highlight and shadow individually by limiting the movement of the snakes based on the known relationship between highlight and shadow [49].

Sonar images are often very noisy, causing the boundaries of shadows to be poorly defined. The process of fitting a superellipse to the data can ignore the noise in the image. Superellipses are a special case of curves that are known in analytical geometry as Lamé curves. The superellipse is able to achieve different shapes such as ellipses, rhomboids, and rectangles by simply changing the squareness of the superellipse function. The resulting superellipse shape is used to classify the object of interest [19].

A unique solution for object detection in sonar is known as change detection. The concept of change detection techniques has been widely researched, but in a limited amount for sonar, mainly because of the different characteristics of the images. The idea is to correlate and compare detected mine-like objects to determine if any changes exist that could lead to classification of the object. The candidate regions considered are only highlights with matching shadows. The last step is determining if the changes detected in the candidate regions imply a mine classification by considering the pixel distribution of each change since man-made objects have normal-like pixel distribution compared to the irregular distribution of natural objects [65].

A final classification algorithm uses a finite state Markov machine, where the set of machine states is isomorphic to the three segments in the region of interest. The machine uses four feature vectors for highlight, shadow, highlight clutter, and shadow clutter, which are compared to four threshold values to classify the region [48].

There are a large number of classification techniques in the literature and only a subset are discussed in this chapter. Regardless of the classification technique used in the

algorithm, the result will be a class decision about each region of interest provided.

3.5 Algorithm Fusion

Some algorithms may be better at detection or classification of certain variations of the target object, while others may be better suited for another variation. This is the logic behind the concept of algorithm fusion, where the classification output from two or more algorithms are combined to improve the overall performance of the system.

The fusion of algorithms can be applied to even the simplest of classification schemes, such as thresholding. In this scenario, a series of algorithms are created, which differ in conservativeness of the threshold value and provide independent errors. Then the algorithms are fused together to reduce the number of false alarms [40].

Alternatively, a single algorithm can be applied to data from multiple aspects and the results can be combined. A positive classification in multiple aspects increases the confidence in the combined result [67].

Or the variation between algorithms could be from using multiple co-registered sonar sensors. The different sensor data is channelized as input to a multi-channel coherence analysis (MCA) framework. The coherence of an object in a certain location between channels increases the confidence in the existence of the object [32].

One fusion algorithm uses three known detection and classification schemes from large organizations, Raytheon, NSWC Coastal Systems Station (CSS), and Lockheed Martin. The fusion starts by grouping regions of interest from each algorithm based on distance. Then the clustered contacts are processed to combine the confidence level for a final thresholding. The two combination techniques considered are the thresholding of the confidence average and two of three positive classifications based on individual algorithm threshold [9].

The basis of algorithm fusion is sound as long as the variations in algorithms

are independent in nature, such that they compliment each other's performance. When appropriately constructed, this technique can easily improve the performance of a system.

Chapter 4

Semi-Synthetic Versus Real World Sonar Training Data for the Classification of Mine-Like Objects

Processing of side scan sonar data has been a highly active field for decades, and specifically the task of detecting mine-like objects (MLOs) is very prominent. Traditionally, this was a manual process involving some post-processing to enhance the sonar image before a skilled operator tediously reviewed each image. Over the years the process has shifted from this manual detection towards automatic target recognition (ATR).

Most of the early ATR research focuses on the shadows and highlights created through the obstruction of sound by any object protruding from the seafloor [49, 38, 64]. In much of the research a model of the target is the basis for the feature, as with the matched filter approach introduced by Dobeck *et al* [17]. This algorithm uses matched filters for various range regions and convolves the filter with the image to detect regions of interest. Some algorithms utilize machine learning type techniques for classification such as neural networks [67, 34, 16], K-Nearest Neighbor [34, 15, 16] and eigen-analysis [50].

The fusion of algorithms has also been considered in the research in various ways.

One method processes high frequency and low frequency sonar images individually and then fuses the classification, either using the matched filter [15] or a wavelet decomposition [67] as the feature extraction method. Other fusion efforts combine co-registered sensor data as input to a classifier [32] or combine the output of known classifiers to determine a fused confidence [9]. One effort uses simple shadow attribute features with basic summing fusion of algorithms as well as the nonlinear Volterra Feature with log-likelihood-ratio-test (LLRT) based fusion rules [2].

The recent advances in autonomous underwater vehicles (AUVs) and sonar capabilities have caused an influx of more advanced computer vision features and machine learning techniques. Some researchers have moved away from the model based approach and have started using local descriptors without target knowledge, such as the Haar-like feature [53]. More state of the art machine learning approaches are being considered, such as boosting [53] and support vector machines (SVMs) [27].

The machine learning techniques of boosting and SVMs require a large training dataset to learn the optimized combination of features. Creating a large dataset can be difficult, time consuming and expensive, which is why it is common to generate synthetic or semi-synthetic datasets for training and testing [53, 24, 11, 1].

The goal of this chapter is to consider the detection capabilities when using such synthetic datasets for training as compared to training on real world examples. For this experiment we use a very popular and simple training algorithm, which uses AdaBoost to select features and creates an optimized cascade of features for classifying windows as MLO or non-MLO [62]. This training algorithm requires a large amount of positive and negative training examples to allow the machine learning element to properly select the best features for the cascade. We run two versions of this experiment, one with the Haar-like feature and another with the local binary pattern (LBP) feature.

The three main contributions of this chapter are as follows:

- Evaluating the use of semi-synthetic datasets for classifier generation in lieu of real world datasets, specifically for classifying MLOs in side scan sonar.
- Quantifying the relative improvement when training on real world data instead of semi-synthetic.
- Providing the datasets used in this research to the academic community. The real world training and testing datasets are of particular significance.[1]

The chapter is organized as follows. Section 4.1 describes each of the training and testing sets and the mechanisms for producing or collecting the data. Then we explain in detail the training algorithm and the two features used for this experiment in Section 4.2. Next, Section 4.3 presents and explains the results from the experiments. Section 4.4 discusses the implications of the experimental results and describes the caveats on the findings. We finish with a conclusion in Section 4.5.

4.1 Training and Testing Datasets

There are some common traits of all of the data in this chapter, which are kept static to reduce the number of variables in the trials. All of the datasets use side scan sonar data captured via REMUS AUVs equipped with two 900 kHz Marine Sonic sensors. The sonar images produced by this sensor are 1024 by 1000 pixels, as are all the images used in this chapter. The vehicles are run in the same locations at a goal altitude of 4 meters to collect 30 meter slant-range data. The data collected for all datasets is limited to this scope; however the framework we describe could be used to consider data collected

[1] http://kastner.ucsd.edu/datasets/barngrover/

Table 4.1. Dataset metrics for real and semi-synthetic. Each includes both training and testing.

	Training				Testing	
	Real	Synth	Synth Large	Synth Source	Real	Synth
Images	975	1,000	5,000	100	920	1,000
Type 1 Observations	426	480	2,531	50	401	486
Type 2 Observations	549	520	2,469	50	519	514
MLO	0	0	0	0	348	76
Positives	975	1,000	5,000	100	1,268	1,076

under broader parameters. The actual mines included in the data are the Type 1 and Type 2 mines shown in Figure 4.1. Note that there are other MLOs present, which are not necessarily designated mine shapes. Table 4.1 shows the metrics of the six different datasets for training and testing.

Figure 4.1. Examples of the inert mines that are placed on the seafloor for various mine related exercises.

The Space and Naval Warfare Systems Center Pacific (SSC-PAC) in San Diego, CA collected all of the data. There are ten different Type 1 mines and seven different Type 2 mines in the various fields used for collection. We present the differences in the creation of the semi-synthetic dataset versus the real world dataset in the following subsections.

4.1.1 Semi-Synthetic Training Datasets

The most prominent problem with research on automatic target recognition in side scan sonar is the lack of labeled datasets. Therefore, to train using machine learning algorithms requiring thousands of examples, it is common to create a synthetic or semi-synthetic dataset. In this section we describe our technique, which attempts to maximize variability with a limited number of positive mine examples.

First we collect positive examples of mine images from real world side scan sonar data. Our pool of positive examples includes 50 images containing Type 1 mines and 50 images containing Type 2 mines. These 100 images make up the dataset we refer to in Table 4.1 as Synth Source, which is used as a baseline for comparison to our semi-synthetic datasets. We then label the mine highlight and shadow individually by choosing points to represent a polygon around each. A trained operator, who is also an author on this chapter, labels the positive targets manually. Next, we collect negative examples

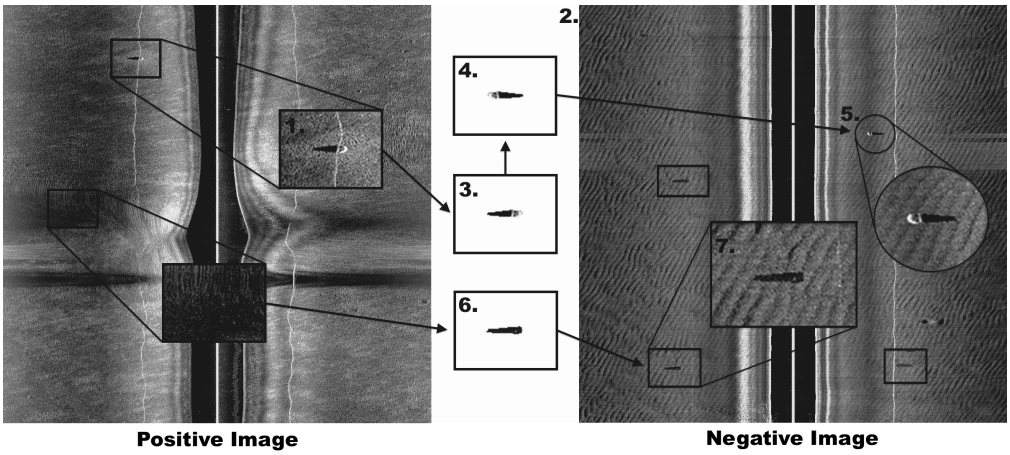


Figure 4.2. The synthetic image creation algorithm. 1. Label the 100 positive examples. 2. Collect the 1,000 negative examples. 3. For each negative image, choose a positive example and capture only the positive pixels. 4. Choose the side of the negative image and a y-location for placement. 5. Copy over the positive pixels. 6. Choose a random location in the positive image and capture negative pixels. 7. Copy the negative pixels to a random location.

from real world side scan sonar data. These are images from the same environment that do not contain a mine. We have 1,000 such negative images for semi-synthetic dataset creation.

The semi-synthetic dataset creation algorithm processes each negative image as a destination image by adding a single mine as well as three patches of background. Figure 4.2 graphically shows the process. For each negative sonar image, one of the 100 positive mine images is selected at random and the polygons of the highlight and shadow are used to copy only the relevant pixels. The side of the negative sonar image on which to place the mine pixels is chosen at random. If the mine is placed on the opposite side of the negative image as compared to the side of positive mine image from which it came, then it is mirrored on the across-track axis to maintain the shadow's proper orientation. Finally, the along-track location for the mine is chosen at random, with the across-track range preserved because of its correlation to the size of the shadow. After all of the randomized parameters are set, the mine can be placed by copying the pixels within the labeled polygons of the positive mine image to the negative image.

In addition, we attempt to account for the potential artifacts introduced through this process by adding randomized non-mine pixels from the positive mine image to the negative, or destination, image. The logic is that any artifacts from the polygon copying process, such as mismatched average pixel intensity or polygon placement on a surface return, are present in these negative portions of our final image as well as in the positive regions. The learning, therefore, does not occur based on any copying artifacts alone. There are potential drawbacks to this which we leave to the discussion in Section 4.4. We extract pixels from random locations in the positive sonar image and place them in the same location of the destination image. The same polygon shapes are used for this background pixel extraction as were used for the positive target pixel extraction to better mimic any artifacts created from the pixel extraction and placement process. We add

three of the negative regions to have more negative than positive examples of any artifacts. The three small rectangles in the "Negative Image" of Figure 4.2 are the randomly placed negative pixels.

There are some flaws to this paradigm for creating a semi-synthetic dataset. For one there is a small number of example positives that must represent a larger sample. The only change for a given mine is the side of the image and the y-location, which is not a lot of variety. The background in close proximity to the mine is another change between semi-synthetic images. However, one flaw is the difference in average intensity from image to image. For example, if the positive sonar image is darker on average then the positive mine stands out even more. In addition, the negative pixels also stand out as darker than the background. The example in Figure 4.2 shows this intensity problem because of the darker background locations of the positive image, which are copied over to the lighter background of the negative image.

Generating one semi-synthetic image takes approximately 0.5 seconds, allowing for very large dataset creation in a reasonable amount of time. All of the negative images are processed to create a semi-synthetic training dataset of 1,000 images, which is referred to as Synth in Table 4.1. The randomization creates a dataset of 480 Type 1 mines and 520 Type 2 mines. One of the benefits of semi-synthetic datasets is the ability to create large numbers of training images. Therefore, we also process each of the negative images five times to create a semi-synthetic dataset of 5,000 images, which is referred to as Synth Large in Table 4.1. This larger dataset contains 2,531 Type 1 mines and 2,469 Type 2 mines. Because of the multiple tiers of randomization in the process, we are able to create a much larger dataset of positive training examples, that contain real world mines in real world backgrounds.

4.1.2 Real World Training Dataset

The ideal training set for creating a classifier for a particular problem is a real world training set. In some cases this is not a difficult endeavor, but in many cases, such as with side scan sonar, the data collection process is expensive and time consuming. The collection of such a large training dataset was a huge undertaking in collaboration with SSC-PAC over the span of a year. This set of images is different from the 100 images used to create the semi-synthetic dataset.

The data collection effort involves various missions in different inert mine fields of San Diego Bay. For a certain mission, the REMUS AUV was programmed to run a reacquire and investigate (R&I) type mission, which is a star-like pattern, over a specific geodetic location. Normally, the R&I mission is designed to collect five to ten meter range data, but in this case was altered to collect 30 meter range data. This type of mission allows us to maximize the number of looks at each mine.

Each mission produces multiple hundreds of side scan sonar images, which must be processed to find the specific mine. Then the mine is labeled with a polygon for the highlight and a polygon for the shadow. This entire labeling process was performed by a trained operator, who is also an author on this chapter, over the course of many months.

The result is a collection of labeled positive images, of which half are designated training and half testing. The real world training dataset used for this experiment contains 975 side scan sonar images, including 426 Type 1 and 549 Type 2.

4.1.3 Testing Datasets

The real world testing dataset is the primary testing dataset for this experiment. It comes from the same collection described in Section 4.1.2, but is a completely separate set from the training. It is also different from the 100 images used to create the semi-synthetic dataset. The real world testing dataset contains 920 side scan sonar images,

including 401 Type 1 and 519 Type 2 as well as 348 other MLOs present in the field.

In addition to the real world test, we have a semi-synthetic testing dataset for evaluating another interesting aspect of this experiment. The same algorithm is used to create this testing set as was used to create the two semi-synthetic training datasets. This means that the same 100 positive images and 1,000 negative background images were used. The only difference is the randomization inherent in the algorithm. The semi-synthetic testing dataset contains 1,000 side scan sonar images, including 486 Type 1 and 514 Type 2 as well as 76 other MLOs present in the base negative images. The synthetic source and the two semi-synthetic classifiers are tested on the real and semi-synthetic testing datasets to analyze the capability of the semi-synthetic training.

4.2 Boosted Cascade Feature Selection

To quantify the benefits of a semi-synthetic training set compared to a real world training set, we must utilize a training algorithm that traditionally requires a large amount of labeled data. We have chosen to use the technique proposed by Viola and Jones [62], which utilizes AdaBoost as a feature selection mechanism to form a cascade of weak learners in stages. Specifically, we have implemented a version of this algorithm under the OpenCV library.

The first step of the algorithm is creating a pool of features for selection purposes. Like the Viola-Jones paper, we use the Haar-like feature in the pool of features for one of our experiment algorithms. The Haar-like feature is based on the Haar wavelet, which does not use intensity directly like the original Haar basis functions [45]. Figure 4.3 shows visualizations of the five types of Haar-like features included in our feature pool.

This feature captures the difference in pixel intensity between designated rectangles in a given location of the considered window. The feature extraction subtracts the sum of the pixels in the white rectangles from the sum of the pixels in the black

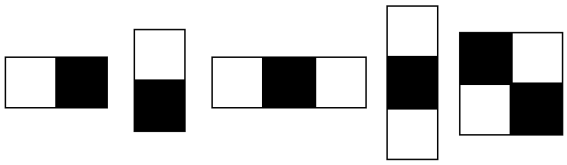
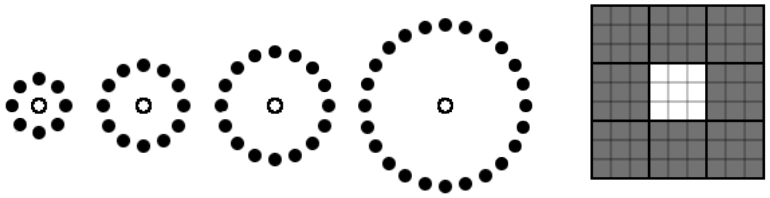


Figure 4.3. The *Basic* group of Haar-like features in the OpenCV library includes these five variations, which we use in this experiment. The sum of the pixels in the white rectangle are subtracted from the sum of the pixels in the black rectangle to calculate the Haar-like feature value.

rectangles, thereby representing the gradient with a single float value. The actual feature calculation is based on a combination of the size of the rectangles and the location of the upper left corner of the feature in the window. The pool of features includes every combination of rectangle and every location of feature possible in the designated window. For the fixed window in this chapter, the result is a pool containing 2,543,145 Haar-like features.

The local binary pattern (LBP) feature is another local type descriptor common to computer vision, which we use for our other experiment algorithm. The LBP feature considers a neighborhood around a center focus pixel and thresholds each pixel intensity compared to the focus pixel, producing a binary representation [42]. Originally the neighborhood was a 3 x 3 region, but has been extended to include multiple sizes [43]. The notation (P,R) represents a given LBP feature, where P is the number of sampling points equally spaced on the circle and R is the radius of the circle. Figure 4.4 (a) shows visualizations of a few possible circular neighborhoods.

This feature captures the texture for a region of the considered window. The feature extraction thresholds each of the pixel locations, represented by the black dots, against the focus pixel, represented by the white dot. Then the resulting binary numbers are concatenated into a string by traveling the circumference of the circle. Finally, the binary pattern is converted to its decimal form, thereby representing the texture with a



(a) Circular Neighhborhood LBP

(b) Multi-Scale Block LBP

Figure 4.4. Visualizations of the two extensions of LBP. Part (a) shows circular (8,1), (12,1.5), (16,2) and (24,3) neighborhoods for LBP calculation. The white dot is the focus pixel and the black dots are neighborhood pattern pixels. Part (b) shows a 3 x 3 multi-scale block LBP. The white square of nine pixels is the focus block and the grey squares are the neighborhood pattern blocks.

single integer.

The LBP feature was extended again to use multi-scale blocks instead of individual pixels to obtain a more macroscopic representation of a local region [35]. In this version the comparison between pixels and the focus pixel is replaced by average pixel intensity of blocks compared with average pixel intensity of a focus block, as visualized in Figure 4.4 (b). We use this multi-scale block LBP feature of the OpenCV library for our second algorithm. The pool of features includes every combination of focus block location and size in the prescribed window. For the fixed window in this chapter, the result is a pool containing 138,645 LBP features.

We use a fixed window due to the nature of the sonar imaging, specifically that a target of a certain size will produce a shadow of varying size based on the altitude of the vehicle and the range to target. This means that differences in the sizes of targets are primarily in the across-track plane and that the shadow part of the target is the only part changing. We are able to choose a fixed window because of the limited scope of that data as described in Section 4.1. An alternative to the fixed window is to use scaling, which is complicated since only the shadow scales and primarily in the across-track plane.

The window size is chosen based on the statistics of the mines labeled in the entire real world training data collected by SSC-PAC. We could have used the largest dimensions from the data or even the geometric maximum dimensions based on the known size of the two mines and the altitude of the vehicle for this dataset. This would produce a larger window, which in turn means more features to consider. Since we are focusing on the transition area between mine highlight and the mine shadow, due to the unknown end of the shadow based on varying length, we do not need a window that includes all possible lengths of target. Therefore, we have chosen to use a standard deviation from the mean to make sure to cover the width of the highlight and a large portion of all shadows. The formulas used to calculate the width and height of the window are Equations **??** and 4.2 respectively.

$$w = \bar{w} + 3\sigma_w + 2m \tag{4.1}$$

$$h = \bar{h} + 3\sigma_h + 2m \tag{4.2}$$

The average width, $\bar{w}$, of the mines, including highlight and shadow, across the entire data library is 42 pixels. The standard deviation of the width, σ_w, is 11 pixels. We also include a margin, m, of two pixels on each side of the mine. The average height, $\bar{h}$, of the mines is 19 pixels, while the standard deviation of the height, σ_h, is two pixels. The same margin, m is used again for the height. The outcome of the statistics analysis is a 79×29 pixel window.

The preprocessing of the training dataset before AdaBoost optimization involves the creation of positive and negative data files based on the labeled location of the MLOs. Since the Haar-like feature includes the location within the window, the positive examples must be aligned such that the highlight portion of the MLO is in the same location in the window. We alter all positive examples to appear as if on the left side of the side scan

sonar image. We also right justify and vertically center the MLO in the window with at least a two pixel buffer. Figure 4.5 shows three examples of positive MLOs aligned in approximately the same location of the window.

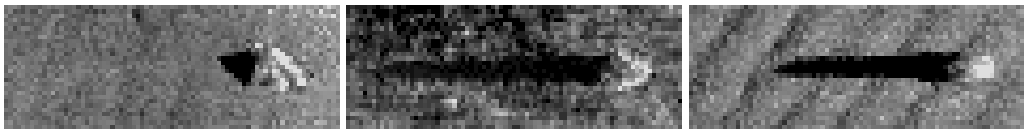


Figure 4.5. These positive MLO examples have varying shadow lengths, but an approximately consistent highlight location within the window.

The negative, or background, image that is provided during preprocessing includes the half of the image that does not contain the MLO. The optimization algorithm utilizes various windows from the background image for training purposes.

In each stage of training, the AdaBoost feature selection process iteratively chooses the next feature, or weak classifier in the jargon of boosting, that best separates the positives from the negatives. The boosting continues to select additional features until the stage is able to achieve certain thresholds for hit rate and false alarm rate. The hit rate is the number of targets correctly classified as positive out of the total number of positive windows considered. The false alarm rate is the number of targets incorrectly classified as positive out of the total number of negative windows considered. In our experimentation, the minimum hit rate is 0.995 and the maximum false alarm rate is 0.5. The boosting terminates when the designated number of stages is met or when the entire classifier passes the acceptance ratio.

The optimized classifier consists of a series of stages each containing features of interest. When the classifier is used to process an image via the OpenCV library, a sliding window approach considers all possible windows of the prescribed fixed window size with an across-track step of one pixel and an along-track step of two pixels. Each stage of the classifier must be passed to label a window as positive. Due to the fine granularity

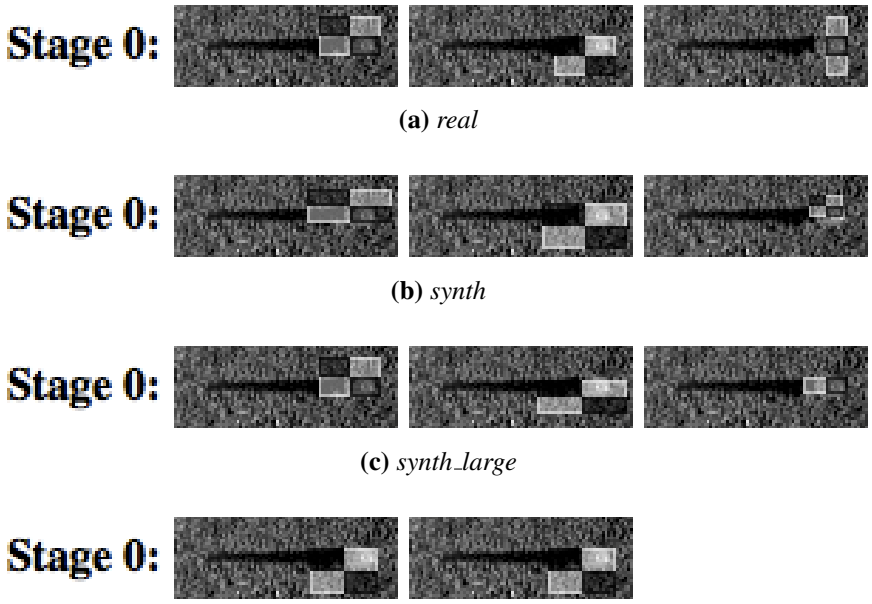
of the sliding window, there are often multiple positive windows in a cluster. There is a minimum neighbors threshold that requires a specific number of windows in a positive cluster for an actual positive labeling. For instance if the minimum neighbors is set to five, then a cluster of five or more positive windows results in one positive label, while four or fewer positive windows results in a negative label.

4.3 Experimental Results

We train on four datasets using two different algorithms for this experiment. The first classifier we refer to as *real* since it was trained on the real world side scan sonar imagery dataset. The second classifier we refer to as *synth* since it was trained on a semi-synthetic dataset. The *synth_large* classifier was trained on a semi-synthetic dataset five times larger than the *real* and *synth* classifiers. Finally, the *synth_source* dataset contains the 100 positive examples, which are the source for the semi-synthetic dataset creation algorithm. The two algorithms described in Section 4.2 use the boosted cascade with the Haar-like feature and LBP feature respectively.

The first algorithm we train on our four datasets uses a pool of Haar-like features. The *real* Haar classifier contains seven stages and a total of 34 Haar-like features. The *synth* Haar classifier has eight stages with a total of 49 Haar-like features and the *synth_large* Haar classifier also contains eight stages, including a total of 73 Haar-like features. The *synth_source* Haar classifier has only four stages with a total of seven Haar-like features. Visualizations of the first stage for each of the four Haar classifiers are shown in Figure 4.6. The feature shows one Haar-like feature from the pool of features presented in Figure 4.3 overlaid on an example mine window. The choice of white or black rectangles does not necessarily correlate to the highlight or shadow in the window, since the feature can be negative or positive to represent the difference in intensity.

The second algorithm in this experiment uses the LBP feature in the pool for



(a) *real*

(b) *synth*

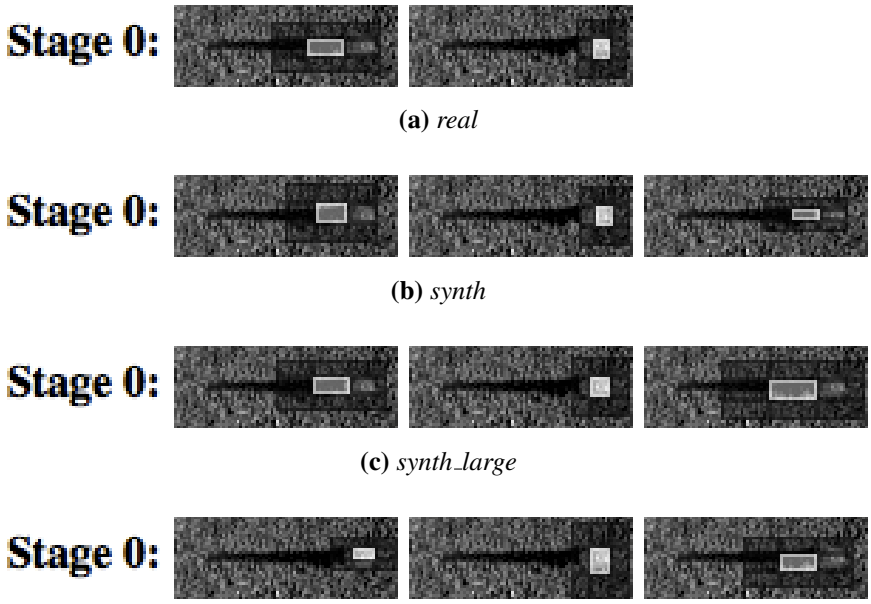
(c) *synth_large*

(d) *synth_source*

Figure 4.6. Visualization of the first stage for each of the four Haar-like Cascade classifiers. Each window shows the Haar-like feature overlaid on an example positive MLO window. The white and black rectangles represent the regions that are compared to calculate the feature value.

boosting. The *real* LBP classifier has 12 stages with a total of 97 LBP features. There are 13 stages for both the *synth* and *synth_large* LBP classifier with 136 and 188 LBP features per classifier respectively. The *synth_source* LBP classifier has 12 stages with only 44 LBP features. Visualizations of the first stage for each of these four classifiers are shown in Figure 4.7. Like the Haar classifiers, each image is a selected LBP feature overlaid on the same example mine window. The white rectangle represents the focus block and the darker rectangles on the perimeter are the comparison blocks.

The efficiency of the algorithms is an important metric to understand for any image processing experiments. The training and processing for these experiments was performed on a 1.7 GHz Intel Core i5 processor with 4 GB 1333 MHz DDR3 RAM. Table 4.2 shows the time taken to train in hours and the processing time of one frame in



(a) *real*

(b) *synth*

(c) *synth_large*

(d) *synth_source*

Figure 4.7. Visualization of the first stage for each of the four LBP Cascade classifiers. Each window shows the LBP feature overlaid on an example positive MLO window. The white rectangles represent the location of the focus block and the black rectangles represent the eight perimeter comparison blocks.

seconds for both Haar and LBP classifiers. The processing time refers to processing a full sonar image. Obviously the type of image will determine how long it takes to process based on how many stages are passed for various windows of consideration.

To understand the performance of the various classifiers, we provide in Figure 4.8 the receiver operating characteristic (ROC) curve for each of the eight classifiers when applied to our real testing dataset. Figure 4.8 (a) shows the curves for the Haar-like classifiers while Figure 4.8 (b) shows the curves for the LBP classifiers. The vertical is the true positive rate (TPR) as with most ROC curves, but on the horizontal we consider false positives per image (FPPI), which is more telling than the false positive rate (FPR) for a sliding window classifier. A FPPI of one means that on average we have one false positive per processed sonar image, which is 1024 by 1000 pixels in these datasets. The

Table 4.2. The classifier efficiencies in terms of training and processing speed for both Haar-like and LBP features.

	Haar		LBP	
	Training (hours)	Processing (secs)	Training (hours)	Processing (secs)
Real	17.55	0.0473	01.60	0.0627
Synth	14.11	0.0499	03.18	0.0801
Synth Large	81.88	0.0508	22.74	0.0812
Synth Source	00.15	0.0415	00.08	0.0682

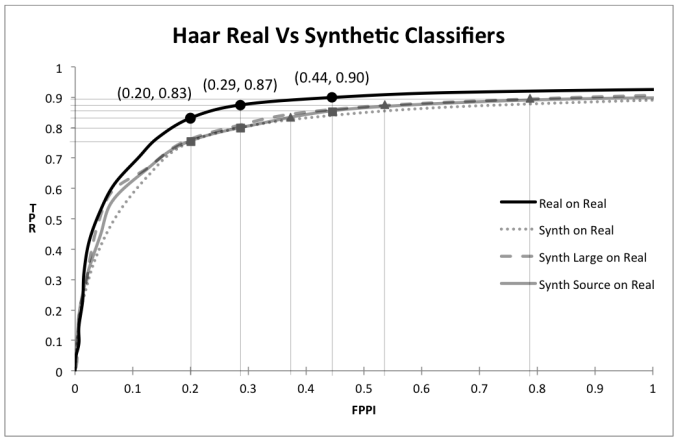
minimum neighbors threshold for clustering positive windows is the variable changed from zero to twenty to create the points on the curve. This means that the actual points on the curves will not have round TPR or FPPI values, but rather have the particular TPR and FPPI for the certain minimum neighbors threshold.

The figures each have example points, shown as black circles, for comparison to the other curves. The triangle points on both figures show the change in FPPI necessary to achieve the same example TPR on other synthetic classifier curves. The square points show the reduction in TPR necessary to maintain the same FPPI on other synthetic classifier curves.

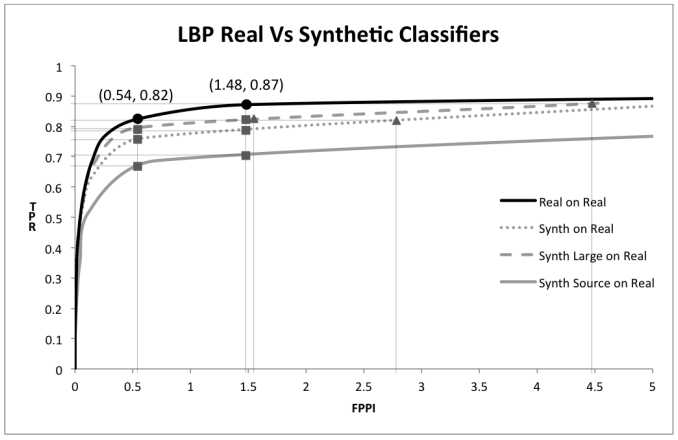
The secondary experiment considered in this chapter is how the synthetic classifiers perform on a semi-synthetic testing dataset compared to on the real testing dataset. Figure 4.9 shows the ROC curves for both Haar in (a) and LBP in (b). The lighter curves show the performance of the semi-synthetic classifiers on the real testing dataset, which are the same curves shown in Figure 4.8, while the darker curves show the performance on the semi-synthetic testing dataset.

4.4 Discussion

Let us start with the visualizations of the first stage for each of the four Haar classifier cascades. As expected the training selects features that emphasize the contrast between highlight and shadow, highlight and background, as well as shadow and background. Many of the features chosen in these cascades are common between the classifiers. Specifically the first two features of Stage 0 are the same feature type in nearly

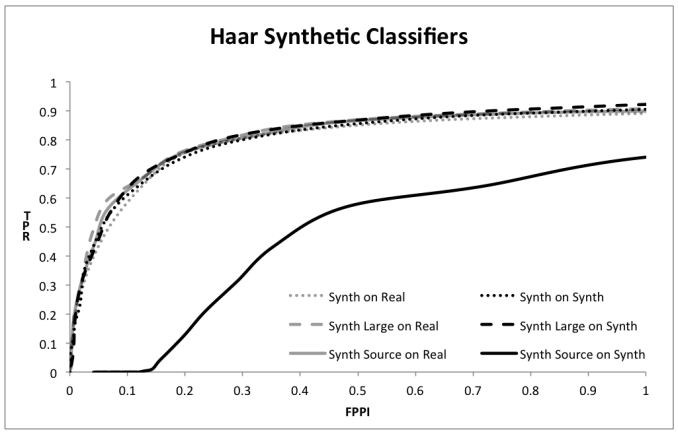


(a) Haar Classifiers

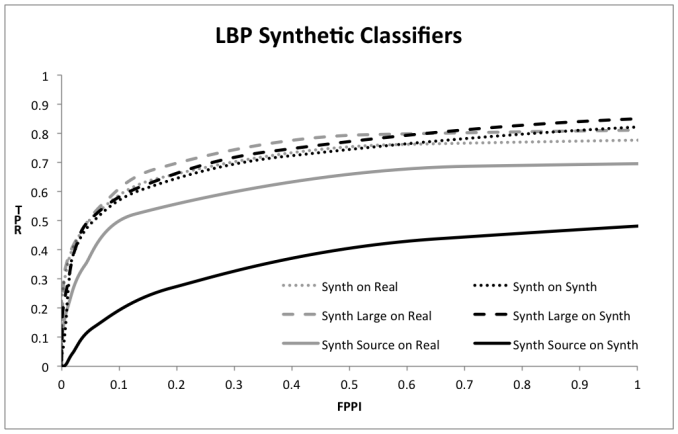


(b) LBP Classifiers

Figure 4.8. The receiver operating characteristic (ROC) curve for the four classifiers when processing the real testing dataset. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per image (FPPI).



(a) Haar Classifiers



(b) LBP Classifiers

Figure 4.9. The receiver operating characteristic (ROC) curve for the *synth*, *synth_large* and *synth_source* classifiers applied to the real and semi-synthetic testing datasets. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per image (FPPI).

the same location with nearly the same size parameters. The exception is the first feature of Stage 0 for the *synth_source* cascade, which is the same as the second feature for all four cascades.

The first stage for each of the four LBP classifier cascades also emphasizes the highlights and shadows compared to the neighborhoods around them. The feature choices are not as consistent as in the first stage of the Haar cascades, but the second feature in

all four is very consistent with a focus block right on the highlight of the example target. This same type of LBP feature is common throughout the stages.

The similarities of the early stages bolster the claim that the semi-synthetic-based classifiers can be used as a proof of concept for the boosted cascade of both Haar-like and LBP features. While the early stages do drastically reduce the number of windows considered from a given image, the later stages do the fine tuned analysis to determine the MLO classification. It makes sense that the obvious differences between a MLO and a non-MLO captured in the early stages would be common among multiple classifiers, while the later stages would have more prominent differences.

The number of features needed to complete the classifier training is interesting. In general, more similarities between the negative and positive examples or more variation within the examples causes a larger feature count in the classifier. For these classifiers, there is an increase in feature count with the size of the training dataset, not because of the increase dataset size directly, but more likely because the additional data provides more variation of positive examples. Also, there is an increase from the real to synthetic classifiers, which implies that the negative and positive examples are more similar and harder to classify in the synthetic training set. This could be caused by some element of the synthetic creation process, such as a high variation within the negative background images used or complications with copying negative pixels to avoid artifact bias.

Analyzing the stages and features of each of the classifiers is one way to grasp the benefit of using semi-synthetic versus real training datasets. Another more telling avenue is to consider the performance on a real testing dataset.

The ROC curves in Figure 4.8 show a couple of important facts. First, as expected, the *real* classifier outperforms the *synth* and *synth_large* classifiers for both Haar and LBP classifiers. This is expected because the semi-synthetic data merely mimics the real data. It is important to note that the improvement with the real classifiers is only

between 4% and 6% over the synthetic classifiers, suggesting the classifiers trained on the semi-synthetic datasets provide a reasonable estimation of the resulting capability of the real classifier.

Furthermore, the *synth_large* classifier slightly outperforms the *synth* classifier. This means that a nominal improvement can be gained from the increase in semi-synthetic training images. However, the *synth_large* classifier takes substantially longer to train, as shown in Table 4.2, when compared to the *synth* classifier for both Haar and LBP. The additional time to train only results in a minor improvement of TPR, just over 3%, for the LBP and an even smaller improvement, less than 2%, for Haar. This is not a large return in accuracy based on the time cost, though the training is a one-time step that might be warranted for the improved capability in some circumstances.

Figure 4.9 shows that the performance of the semi-synthetic classifiers on the semi-synthetic testing dataset is nearly the same as the performance on the real testing dataset for both Haar and LBP. For the Haar classifiers there is only about 1% difference of TPR between the real and synthetic test sets for *synth* and *synth_large*. There is a bit more separation in performance using the LBP feature, with about 3% difference in TPR. The similarity of performance for the synthetic classifiers on both testing datasets is important because it means that training and testing on semi-synthetic data can appropriately estimate how these classifiers perform on a real testing dataset.

The exception is the *synth_source* classifier, which performs much worse on the semi-synthetic testing dataset. This is a result of a specific bottom type containing sand ripples, which is present in the synthetic data but is less prevalent in the real testing data. This sand ripple bottom type is not in the *synth_source* training set, causing a large number of false positives when tested on the synthetic data compared to the real data. The *synth* classifier does not have the same problem when tested on the synthetic data because it is trained on images including the sand ripple bottom.

To verify that the sand ripples are causing the large number of false positives, we test on a subset of the synthetic testing set with some images removed. We remove sixteen images with prominent sand ripple regions from the synthetic testing set and process this new testing set. There is a strong improvement just from removing a small number of sand ripple bottom images.

This exception emphasizes the need to have large comprehensive datasets for training when using such machine learning algorithms as boosting. The small number of training images is part of the problem, but this just highlights the poor performance resulting from a disproportionate variety in the training data compared to the testing data.

The major take-away from these experimental results is that semi-synthetic training and testing in the absence of real data can provide expectations for the performance when trained and tested on real data. Furthermore it can lead towards decisions on the viability of one feature over another. For instance, we could correctly estimate from the semi-synthetic classifiers on semi-synthetic testing data that the Haar classifier is better for this dataset than the LBP without needing the real data.

The data labeling is one problem with this research. We use one trained operator to label all of the positive MLOs for synthetic dataset creation as well as for the real training and testing datasets. This is a manageable approach for a small experiment, but an intra- and inter-operator study would be an interesting experiment in the future.

The semi-synthetic dataset creation process has some shortcomings. There are only 50 unique mine examples of each mine type used to create the two different sizes of synthetic training datasets. This is a result of available data at the time, not a chosen parameter. For a given mine type, there is a limited amount of variation between observations when the data is captured at a constant altitude and frequency. Our 50 observations per mine type does not cover all of the variations, but given the size of the mine at this range, it covers a large amount of variation. We acknowledge that it would

be worthwhile to create synthetic datasets with the larger number of unique positives now available.

Another problem with the semi-synthetic dataset creation process is that the copying of background pixels to the synthetic target image may create MLOs based on differences in average pixel intensity between images. One could consider normalizing the images so that they are similar in average pixel intensity to avoid this problem. Also, it would be interesting to remove from the creation process the step of copying the background pixels and then compare the results. This may improve the classifier, but it will be hard to quantify if this is a result of a copying artifact in the synthetic data versus removal of problem negative polygons.

There are limitations to this experiment based on the limited training algorithms considered and the singular semi-synthetic data creation scheme. It would be interesting to consider more training algorithms and other synthetic and semi-synthetic data in future experiments to more broadly quantify the use of automated data in the absence of real data.

4.5 Conclusion

The experiments considered in this chapter show the benefit of using a semi-synthetic dataset for the training and testing of data hungry machine learning algorithms, such as the boosted cascade using either the Haar-like feature or the LBP feature, when a real dataset is not available. The experimental results show that semi-synthetic classifiers perform within 1% and 3% for Haar and LBP respectively when tested on semi-synthetic data versus real data. The results also show that the semi-synthetic classifiers perform only 4% to 6% worse than the real classifiers when tested on real data. Combined, the results from the two experiments mean that only using semi-synthetic data for training and testing when real data is not available can provide insight into the capability of an

algorithm.

Despite the limited scope of the experimentation in this chapter, the results provide a foundation for comparison between synthetic and real training in terms of side scan sonar imagery and mine-like object classification. And, in the context of this chapter, it is clear that there is a benefit in using semi-synthetic datasets for training in the absence of available real world data.

This chapter, in full, is a reprint of the material as it appears in IEEE Journal of Oceanic Engineering, Barngrover, Chris; Belongie, Serge; Kastner, Ryan, 2014. There are small changes in format and phrasing as a chapter within this larger paper. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Multi-Feature Selection Framework for the Detection of Mine-Like Objects in Side Scan Sonar Imagery

Object detection and classification in underwater environments is a difficult yet promising endeavor, which can lead to finding and removing mines, vastly improving the safety of the military. Currently for this task, trained operators analyze images to find the desired targets manually, which is very time consuming. Automating these tasks would be quite valuable, augmenting the capabilities of operators in order to clear larger regions of mines in shorter time periods. However, these tasks are not easy to solve in the dynamic and complex underwater environment.

The research on methods for automatically detecting mine-like objects (MLOs) in side scan sonar is vast. In the past, the object detection algorithms relied mostly on models of the target and in some cases they utilized limited versions of machine learning for improving a classifier. A majority of the approaches focus directly on the shadows and highlights created by the occlusion of sound propagation, and the reflection of energy from the protruding MLO, respectively [38, 49, 33]. In much of the research a model of the target is the basis for the feature, as with the matched filter approach introduced by Dobeck et al. [17]. This algorithm uses matched filters for various range regions and

convolves the filter with the image to detect regions of interest.

The concept of fusion to achieve improved performance has shown promise, whether it be the combination of results from a single classifier on data from multiple frequencies [15, 67], the combination of co-registered sensor data as input to a classifier [32], or the combination of outputs from known classifiers to increase or decrease confidence [9]. There have been some machine learning approaches applied to the regions of interest, including neural networks [67, 33] and eigen-analysis [50].

The above approaches primarily focus on a model of the target as a basis for object detection. The recent advances in AUVs and the side scan sonars they utilize has led to the application of various computer vision and machine learning techniques for training detectors. These techniques use features that focus on a local description within a window of the sonar image and large training datasets for the machine learning algorithms.

One prominent algorithm, introduced by Viola and Jones, uses the Haar-like feature and a feature selection process by the AdaBoost machine learning algorithm [62]. The so called Viola-Jones algorithm has shown great capability for speed and efficiency. The original paper describes this as a face detector, but it has since been applied to many other targets [39, 36, 30]. Sawas and Petillot applied this Viola-Jones technique to MLO detection in sonar, using an AdaBoost variant known as GentleBoost for feature selection of Haar-like features to train individual classifiers for each mine type [54].

Another powerful local feature descriptor is the Scale-Invariant Feature Transform (SIFT) feature, which is the predecessor to the Speeded Up Robust Feature (SURF). These features are commonly used in interest point extraction for localization techniques such as Simultaneous Localization and Mapping (SLAM) [4]. Hollesen et al. considered the SIFT feature with a support vector machine (SVM) to create a confusion matrix for MLO detection [27].

5.0.1 Our Approach

The model based approaches, which focus on the shadows, have been very heavily researched and are powerful features for this task. The more recent exploration of local features such as the Haar-like feature and the SIFT feature show promise for MLO detection. The current gap is the combination of these newer local features and the more well documented model features. In this chapter we propose a multi-feature selection framework, which allows the GentleBoost feature selection process to choose from a feature pool including multiple types of features. Our approach shows that a combination of features in a cascade can provide capability improvements over a single-feature selection process.

In developing our multi-feature algorithm we experimented with many local features, but this chapter focuses on six features, including four local features, one model feature that focuses on the shadow, and one mixed feature based on the Haar-like feature but limited by the model of the MLO. We explain these features further in Section 5.2. Nearly all of the multi-feature classifiers produced from this selection framework show an improvement over the best single-feature classifier.

In the general computer vision community, there have been multiple adaptations of the single-feature selection algorithm. Some of the simple adaptations alter the Haar-like pool [36] or use the other features such as the Histogram of Oriented Gradients (HoG) [70]. Then there have been some efforts that utilize a feature pool containing more than one feature similar to our approach. Zhang et al. [69] describe a boosted cascade that selects features from a pool of both SIFT and space context features, which Wang et al. [63] later adapted for gender classification via faces. Feris et al. applied this multiple feature concept to urban surveillance by creating a pool of Haar-like features from four different color channels [20].

There are four major contributions of this chapter, which we outline here:

- This is the first proposition of applying a multi-feature selection process for the detection of mine-like objects in side scan sonar imagery.

- This chapter builds on the work of Sawas and Petillot [54] and Hollesen et al. [27] to consider other prominent computer vision features with GentleBoost feature selection as a baseline for the multi-feature work. These features and their results on our test dataset are presented in Section 5.2.

- The multi-feature selection framework is structured such that any dataset and any group of features can be used for training and classification. Unlike the work of Sawas et Petillot [54] our feature selection process considers features in addition to the Haar-like feature. The concept is similar to the multiple feature pools employed by Zhang et al. [69] and Feris et al. [20], except the features are chosen based on preliminary research into the features capability for detecting MLOs in side scan sonar imagery. Also we consider many combinations rather than just one combination.

- The optimal combination of feature types selected by the multi-feature selection framework introduced in this chapter combines the proven model-based shadow feature type with the local Haar-like feature and the interest point speeded up robust feature. The resulting *HAAR + SURF+ SHADOW* classifier improves the true positive rate by 12.69% up to 88.56% at approximately 3.0 false positives per km^2 over the single *HAAR* classifier. Under other parameters, the classifier improves true positive rate by 3.79% while reducing false positives per km^2 by 1.74 compared to the *HAAR* classifier.

The chapter is organized as follows. We begin with an in depth explanation

of boosted feature selection in Section 5.1. Then in Section 5.2 we describe the pool of features, including the designed window size and the six feature types. Section 5.3 introduces the data used in this chapter. The experimental results are presented in Section 5.4 and we conclude with Section 5.5.

5.1 GentleBoost Feature Selection

In order to understand the multi-feature selection suggested in this chapter, it is important to understand the preceding variations of boosting. The AdaBoost algorithm, introduced by Freund and Schapire [22], iterates over training data (x_i, y_i) where x_i is the data and $y_i = \{-1, 1\}$ is the label for positive or negative. In each iteration t, the algorithm trains a weak learner to update the corresponding weight, w_i. The weak learner provides a weak hypothesis, $h_t \rightarrow \{-1, 1\}$ across the training data with some error, ε_t. Then the weight, w_i, is updated based on the error, ε_t, which is shown in Equation 5.1, where the result is normalized such that $\sum_i w_i = 1$. This adaptation component was the breakthrough of the AdaBoost algorithm over previous boosting algorithms.

$$w_i \leftarrow w_i \exp\left[y_i h_t(x_i) \times \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\right] \tag{5.1}$$

The next iteration of AdaBoost, proposed by Schapire and Singer, is called Real AdaBoost, while the original algorithm became known as Discrete AdaBoost [55]. This is because the new version uses real value confidence predictions rather than a discrete {-1,1} hypothesis for the label of a datum. The Discrete AdaBoost uses a hypothesis $h_t(x) \rightarrow \{-1, 1\}$ but for Real AdaBoost the hypothesis becomes $h_t(x) \rightarrow R$, where the sign of the real value determines the classification prediction and $|h_t(x)|$ gives the confidence of the classification. The weak learner hypothesis is a probability function $h_t(x) = P(y_i = 1|x)$, which results in a slight change to the weight update function, as

shown in Equation 5.2.

$$w_i \leftarrow w_i \exp\left[y_i \times -\frac{1}{2}\log\left(\frac{h_t(x_i)}{1-h_t(x_i)}\right)\right] \tag{5.2}$$

The GentleBoost variation of boosting, introduced by Friedman et al., builds on the Real AdaBoost algorithm, but uses adaptive Newton steps to update the classification rule rather than the more volatile log-ratio update [23]. This allows for smaller changes to the weak learner at a given update step compared to the potentially large updates in Real AdaBoost. The classification output is still a real value where the sign gives the classification and $|h_t(x)|$ gives the confidence. The weight update function is altered again, as seen in Equation 5.3. Instead of taking the log-ratio of the probabilities given by the weak learner hypothesis, $h_t(x)$, the more stable difference is used.

$$w_i \leftarrow w_i \exp[y_i \times (1-2h_t(x_i))] \tag{5.3}$$

The Discrete AdaBoost algorithm was altered by Viola and Jones to make it a feature selection process by restricting the weak learner considered in each iteration of training to only depend on one feature [62]. This means that each iterations selects a feature since the weak learner is just one feature in this case. This result is a boosting algorithm that can iteratively create strong classifiers composed of individual features.

The GentleBoost feature selection algorithm applies the feature selection concept introduced by Viola and Jones [62] to the GentleBoost algorithm proposed by Friedman et al. [23]. This is the implementation used by Sawas and Petillot for the detection of mine-like objects [54]. It is also the implementation of the boosting feature selection that we use in this chapter.

The feature selection process operates by iteratively selecting new weak learner

features until achieving a target hit rate and false alarm rate. The hit rate is the number of targets correctly classified as positive out of the total number of positive windows considered. The false alarm rate is the number of targets incorrectly classified as positive out of the total number of negative windows considered. When a combination of selected features is able to find 99.5% of the targets and only have at most 50% false positives then the algorithm can exit the round of training, having created one classifier that acts as a stage. The algorithm will continue this process adding stages to the cascade until the prescribed number of stages has been created, fourteen in this research, or the overall cascade achieves an acceptance ratio.

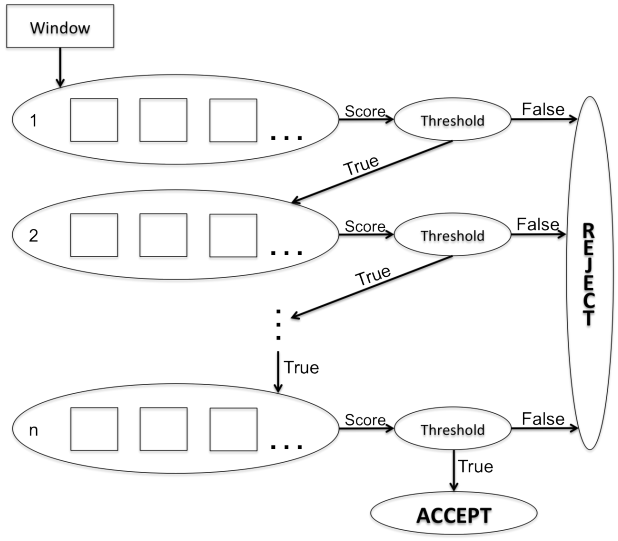


Figure 5.1. The schematic view of a cascaded classifier. Each classifier stage produces a score that is tested against a threshold to determine if the stage is passed. If any stage threshold is not passed then the window is rejected. If all stages are passed then the window is accepted.

The result of the training step is an optimized classifier consisting of stages, each containing a classifier composed of features. Figure 5.1 shows a schematic view of this cascade concept. The individual stage classifier is the oval where each square within is a feature. The stage provides a score, which is thresholded to determine whether to reject

or continue to the next stage. The score for a stage classifier is the strong hypothesis, $H(x)$, given by Equation 5.4, where m is the feature and M is the total number of features in the stage.

$$H(x_i) = \text{sign}\left[\sum_{m=1}^{M} h_m(x_i)\right] \tag{5.4}$$

For the current window being processed, if all stages are passed then the window is accepted as a positive window.

The boosted cascade classifier uses a sliding window approach, meaning that nearly all possible windows are considered, starting with the upper left corner and using a horizontal step of one pixel and a vertical step of two pixels. The sliding window considers many overlapping regions causing multiple positive windows to result in a cluster. The processing component has a *minimum neighbors* threshold to require a certain number of positive windows in a cluster for a positive acceptance. For instance if the *minimum neighbors* is set to five, then the image processing must produce at least five positive windows in a cluster in order to provide a positive label, while four or less positive windows results in a negative label.

The important point in this explanation of the boosting feature selection concept is that the algorithm, specifically in our research the GentleBoost algorithm, selects features from a designated pool of features. The actual feature calculation is abstracted from the boosting and only the resulting feature score is considered. In other words a specific feature applied to a specific window will produce some arbitrary floating point value, while some other feature will produce a separate floating point value. The boosting uses this value combined with a label of positive or negative in order to train, leaving the feature calculation abstracted. This means that the boosting is separate from the feature pool creation process, which we will discuss in Section 5.2. We take advantage of the feature-agnostic component of the feature selection algorithm in order to create

our multi-feature selection framework.

5.2 Pool of Features

In this section we discuss the various features that we consider for the creation of classifiers under the single-feature and multi-feature selection frameworks. In the first subsection we introduce the standard window size used for this research and the reasons for choosing a fixed window size vice scaling. The remaining subsections describe each of the six features and the implementation.

5.2.1 Window Size

The window size for training and testing is essential to this algorithm. For some of the features it will predicate the number of variations added to the pool. Since the Viola-Jones algorithm was designed to detect faces, it was assumed that the face could be multiple sizes in pixels and therefore the algorithm considers many window sizes by scaling them all to a standard size for processing. Due to the nature of the sonar image and the consistency of the range for the sonar dataset used for this research, the scaling capability is not necessary for this application. Instead we choose one standard window size for training and testing.

The window size is chosen based on the statistics of the mines labeled in the training dataset, which will be described in Section 5.4. From the labeled data we calculate the mean and standard deviation in order to maximize the number of the mines included without too large a window to process. The formulas used to calculate the width and height of the window are Equations 5.5 and 5.6 respectively.

$$w = \bar{w} + 3\sigma_w + 2m \tag{5.5}$$

$$h = \bar{h} + 3\sigma_h + 2m \tag{5.6}$$

The average width, $\bar{w}$, of the mines, including highlight and shadow, across the entire data library is 42 pixels. The standard deviation of the width, σ_w, is 11 pixels. We also include a margin, m, of two pixels on each side of the mine. The average height, $\bar{h}$, of the mines is 19 pixels, while the standard deviation of the height, σ_h, is two pixels. The same margin, m is used again for the height. The outcome of the statistics analysis is a 79×29 pixel window.

While this window size analysis is based on our dataset, it does generalize to future data. This is because the variable of the altitude, which affects shadow size, is consistent in order to collect at this data range. Also, because the data covers a wide variety of angles and x-locations for looks at the two mine types considered. A similar window size analysis might be necessary if different mines or ranges are considered.

5.2.2 Haar-Like Feature

The Haar-like feature described by Viola and Jones is based on the Haar wavelet and captures the difference in overall pixel intensity between neighboring regions [62]. Different shapes and locations of the features can represent various characteristics of a region without needing prior knowledge about the model of the target. Figure 5.2 shows visualizations of the five Haar-like features we consider. The feature value is the difference between the total intensity of the white regions and the black regions in the visualization. There are many variations of the Haar-like feature discussed in the literature, but we focus on this basic subset for this research.

The calculation of these Haar-like features, summing so many pixels and then subtracting, can be processor intensive when training and feature selecting from a pool of millions of variations. The solution is the implementation of the integral image, which is generated once at the start of processing an image. The integral image is formulated

from the original image, where each pixel contains the sum of the pixel intensities for the rectangle to the upper left of that point.

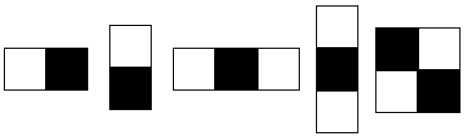


Figure 5.2. The visualization of the five Haar-like features that we use in this experiment. The sum of the pixels in the white rectangle is subtracted from the sum of the pixels in the black rectangle to calculate the Haar-like feature value.

Figure 5.3 shows a zoom view of a MLO in a side scan sonar image. The integral image processed from this image would contain at point 1 the sum of the pixel intensities in rectangle A. It follows then that the value at point 2 would be the sum of the pixels in rectangles A and B. The speed of this solution becomes apparent when we attempt to calculate the sum of the pixels in a rectangle over the shadow of the mine for part of a Haar-like feature, which in Figure 5.3 is $D = 4 - 2 - 3 + 1$. This simple arithmetic is much faster when processing a large number of Haar-like feature over a large image.

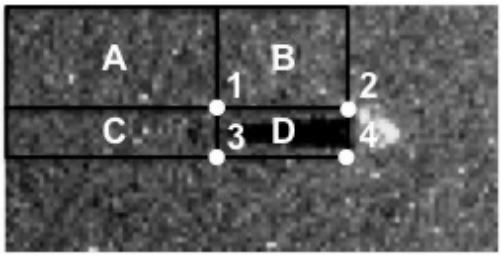


Figure 5.3. Overlay of integral image calculation on a MLO in a sonar image. Each point contains the sum of the pixels in the rectangle to the upper left. For example point 1 contains the sum of the pixels in the rectangle A while point 2 contains the sum of the pixels in rectangles A and B.

Each feature is comprosed of the location of the feature within the window and the size and variation of the rectangles. We add to the pool every combination of size and

location of the five types within the considered window size. For the chosen window size and the five types, there are 2,543,145 Haar-like features in the pool of features.

5.2.3 Histogram of Oriented Gradients Feature

The histogram of oriented gradients (HOG) feature was introduced by Dalal and Triggs for the application of pedestrian detection [13]. The feature is intended to capture the local edge directions as a simple representation of shape. The distribution of the edges, which are represented by the intensity gradients, provides a characterization of shape without direct knowledge of the corresponding edge positions.

A HOG feature begins with a *block* location and splits this region of the image window in to a set number of *cells*. For each *cell* the pixels are used to accumulate the the local 1-D histogram of gradient detections with a set number of *bins*. Based on the implementation by Dalal and Triggs [13], this implementation splits a *block* in to four 8×8 *cells* and uses nine *bins* in the histogram. Figure 5.4 shows an example *block* location with the four 8×8 *cells*. The bottom right *cell* shows a visualization of the nine *bins* represented as gradient directions.

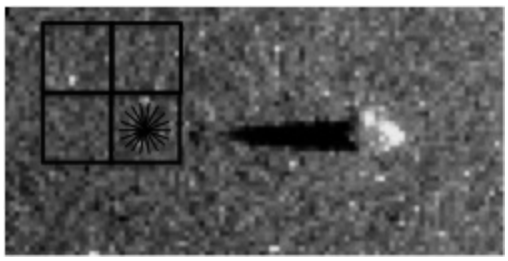


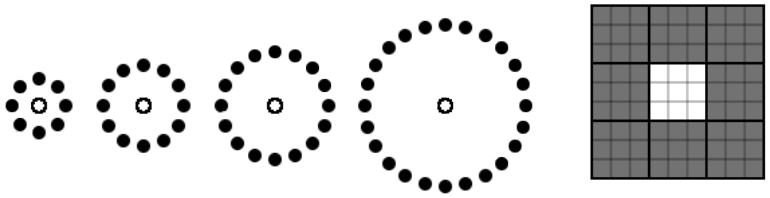
Figure 5.4. The visualization of the histogram of oriented gradient (HOG) feature. This shows an example *block* location and its division in to four 8×8 *cells*. The bottom right *cell* shows a visualization of the nine orientation *bins* in the histogram.

A feature in the context of this implementation is a bin in one particular cell so for each feature region there are 36 features. With our fixed window size and our *cell*

dimensions and number of *bins* there are 4,032 HOG features in the pool of features.

5.2.4 Local Binary Patterns Feature

The local binary pattern (LBP) feature is another local type descriptor common to computer vision, which considers a neighborhood around a center focus pixel and thresholds each pixel intensity compared to the focus pixel, producing a binary representation [42]. In the original feature, the neighborhood used was a 3 x 3 region, but it has been extended to include multiple sizes [43]. The notation (P,R) represents a given LBP feature, where P is the number of sampling points equally spaced on the circle and R is the radius of the circle. Figure 5.5 (a) shows visualizations of a few possible circular neighborhoods.



(a) Circular Neighhborhood LBP

(b) Multi-Scale Block LBP

Figure 5.5. Visualizations of the two extensions of LBP. Part (a) shows circular (8,1), (12,1.5), (16,2) and (24,3) neighborhoods for LBP calculation. The white dot is the focus pixel and the black dots are neighborhood pattern pixels. Part (b) shows a 3 x 3 multi-scale block LBP. The white square of nine pixels is the focus block and the grey squares are the neighborhood pattern blocks.

This feature captures the texture for a region of the considered window. The feature extraction thresholds each of the pixel locations, represented by the black dots, against the focus pixel, represented by the white dot. Then the resulting binary numbers are concatenated into a string by traveling the circumference of the circle. Finally, the binary pattern is converted to its decimal form, thereby representing the texture with a

single integer.

The LBP feature was extended again to use multi-scale blocks instead of individual pixels to obtain a more macroscopic representation of a local region [35]. In this version the comparison between pixels and the focus pixel is replaced by average pixel intensity of blocks compared with average pixel intensity of a focus block, as visualized in Figure 5.5 (b).

In this research we consider this multi-scale block LBP feature. The pool of features includes every combination of focus block location and size in the prescribed window, which produces 138,645 LBP features for our fixed window size.

5.2.5 Speeded Up Robust Feature (SURF)

The Speeded Up Robust Feature (SURF) [6] is a scale and rotation invariant interest point detector and descriptor based on the Scale-Invariant Feature Transform (SIFT) [37], but with vast speed improvements and more robustness against different image transformations. This feature is an interest point detector and each interest point has a descriptor representing the feature. The detector finds interest points at corners and edges, which are prominent on the shadow and sometimes the highlight of the MLO. The scale and rotation invariance of SURF is not necessary for this target, but the interest point detector is a unique feature to consider for this sliding window feature selection paradigm. The concept we describe here for using the this interest point detector could be applied to other similar features.

The interest point detector is based on the determinant of the Hessian matrix to maintain the scale invariance goal. The result of this is that a Hessian threshold can be used to determine if an interest point should be considered. A smaller Hessian threshold, such as 1-500, will cause more interest points to be found per image, while a higher threshold, such as 4000, may only capture a few interest points per image. The SURF

descriptor is based on the sum of Haar wavelet responses in a given region around a detected interest point.

Since SURF is based on an interest point detector, it begins by determining all interest points in an image and then continues by calculating the SURF descriptor for each point. This does not coincide well with the sliding window approach of the boosted cascade classifier. In order to preserve the interest point nature of SURF, we train a predictor using the same single-feature selection GentleBoost algorithm [23] to estimate if a given SURF descriptor represents a positive or negative interest point. The descriptor and interest point information are the features provided to the boosting along with positive and negative labels. All SURF features with a Hessian greater than 500 are considered in the training. The result is a binary tree predictor that produces a score for a SURF point based on the provided descriptor and point information.

Our SURF based feature includes a Hessian threshold and predictor threshold combination. For the given combination all interest points over the Hessian threshold are predicted using the predictor threshold. The feature score is the number of interest points predicted to be positive within the window.

Our pool of features includes every SURF based feature combination of Hessian threshold from 300 to 5000 in increments of 100 and predictor threshold from -1.0 to 2.0 in increments of 0.05. The result is 2,820 SURF features in the pool of features.

5.2.6 Shadow Threshold Feature

The shadow threshold feature is a simple feature that is closer to the model approaches of the past. The goal of this feature is to capture the nature of sonar images; namely that all protruding objects from the sea floor produce an acoustic shadow. There are many shadows in a sonar image, so this feature alone would surely create a large number of false positives. However, when combined with other features it should help

alleviate false positives where no shadow exists yet there exists some pattern that matches another feature.

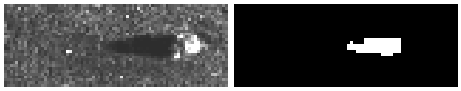


Figure 5.6. The right image shows the white representing the shadow based on binary connected component estimation via thresholding the left image at a pixel intensity of one. The connected component with the most area is considered further as the shadow object for this window.

To find the shadow, the window is first thresholded at a certain pixel intensity and then the most likely connected component is estimated based on area. Figure 5.6 shows a sample region with a MLO and the binary connected component estimation of the shadow using a pixel intensity threshold of one. Once this connected component is isolated we estimate its border as a polygon using the method introduced by Suzuki et al. [58].

The connected component and the estimated border are used to create the series of features that actually comprise this feature set. These features include width, height, ratio of width to height, and area. Also the center of the connected component is estimated and used to provide additional features including the absolute X location in the entire image, the relative X location in the window, the relative Y location in the window, and the altitude of the vehicle at the time this location of the image was captured. We are able to use the altitude at the center of this connected component because we actually process the raw sonar image, as explained in Section 5.3.

The final group of features related to the shadow is based on shape matching between the estimated border and a set of 54 predefined shadow shape templates. We use three different shape matching functions each based on the Hu moment invariants [28]. In the following matching equations A is the first shape, such as the estimated border of the

connected component, and B is the second shape, such as a predefined shape template.

The three shape matching equations are

$$I_1(A,B) = \sum_{i=1\ldots7} \left| \frac{1}{m_i^A} - \frac{1}{m_i^B} \right| \tag{5.7}$$

$$I_2(A,B) = \sum_{i=1\ldots7} |m_i^A - m_i^B| \tag{5.8}$$

$$I_3(A,B) = \max_{i=1\ldots7} \frac{|m_i^A - m_i^B|}{|m_i^A|} \tag{5.9}$$

where

$$m_i^A = \text{sign}(h_i^A) \cdot \log h_i^A \tag{5.10}$$

$$m_i^B = \text{sign}(h_i^B) \cdot \log h_i^B \tag{5.11}$$

and h_i^A and h_i^B are the Hu moments of A and B, respectively. The actual shape features are based on the similarity scores produced via these equations.

The first feature in this group is the best match result, followed by the average of the best two, the average of the best three, and so forth up to the average of the best 20 match results. The result is 20 shape features per matching variation plus the eight regular features, along with thresholding at pixel intensities of zero to ten, creating a pool of 748 shadow features.

5.2.7 Sonar Feature

The last feature considered, which we call the sonar feature, is a variation of the Haar-like feature based on knowledge about the sonar image and the signature of an object protruding from the sea floor. The focus of this feature is the relationship between the background sea floor, the highlight region, and the shadow region and therefore is composed of these three regions. Figure 5.7 shows an example sonar feature overlaid on

a window containing an MLO. The shadow region is shaded black, the highlight region is shaded white with a white outline, while the background is the remainder of the window and is shaded white.

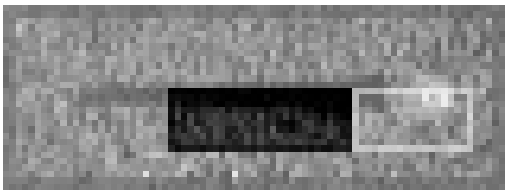


Figure 5.7. Overlay of the sonar feature on a MLO in a sonar image window. The dark regions is the shadow portion of the feature and the white outlined region is the highlight. The remainder of the window, which is also shaded white, is the background region of the feature. The feature is calculated by summing the differences between the background and shadow, highlight and shadow, and highlight and background.

The calculation of the feature uses the integral image just as the Haar-like feature does. First the intensity of the shadow region s and the highlight region h are calculated as for a Haar-like feature. Then the background region b is calculated by taking the intensity of the entire window and subtracting out the shadow and highlight intensities. The resulting feature value sums the differences between each region by expected relationship, namely that the shadow should be darker than the background and the highlight and the highlight should be brighter than the background. Equation 5.12 shows the calculation of the sonar value v.

$$v = (b - s) + (h - s) + (h - b) \tag{5.12}$$

The variation of the feature is the location and size of the shadow and highlight regions within the window. Given all possible combinations of these variables there are 646,866 sonar features in the pool of features.

5.3 Dataset

The data used for this chapter was collected by the Space and Naval Warfare Systems Center Pacific (SSC-PAC) in San Diego, CA. The data was collected from multiple REMUS vehicles equipped with two Marine Sonic side scan sonars operating at a frequency of 900kHz to capture 30 meter range data based on a four meter altitude. The resulting sonar image, which is a proprietary Marine Sonic image format, is 1024 by 1000 pixels and contains sonar intensity information as well as sensor information such as altitude. In this research our training and testing are both done directly on the raw sonar data.

(a) Type 1. **(b)** Type 2.

Figure 5.8. Examples of the inert mine types contained in the datasets used for this experiment. These objects sit on the sea floor protruding up from the bottom.

The test mines in this data include ten truncated cones (Type 1) and seven stealth wedges (Type 2). Examples of these two mine types are shown in Figure 5.8. Multiple passes of the mine locations produce the many looks included in our data, which is split into a training dataset of 975 images containing 975 target mines and a testing dataset of 920 images containing 1,268 target mines.

The data we have collected and utilized for this research is considered an easy dataset. This is the type of environment that is currently considered in real world mine clearing situations. There is strong interest in research involving the more cluttered sea bottoms, but our research focuses on the current need for a capability on relatively

easy, very shallow water (VSW) environments. Despite this, the data does have slightly complex bottoms and some clutter to note. Figure 5.9 shows some examples of variation in complexity, image quality, and clutter.

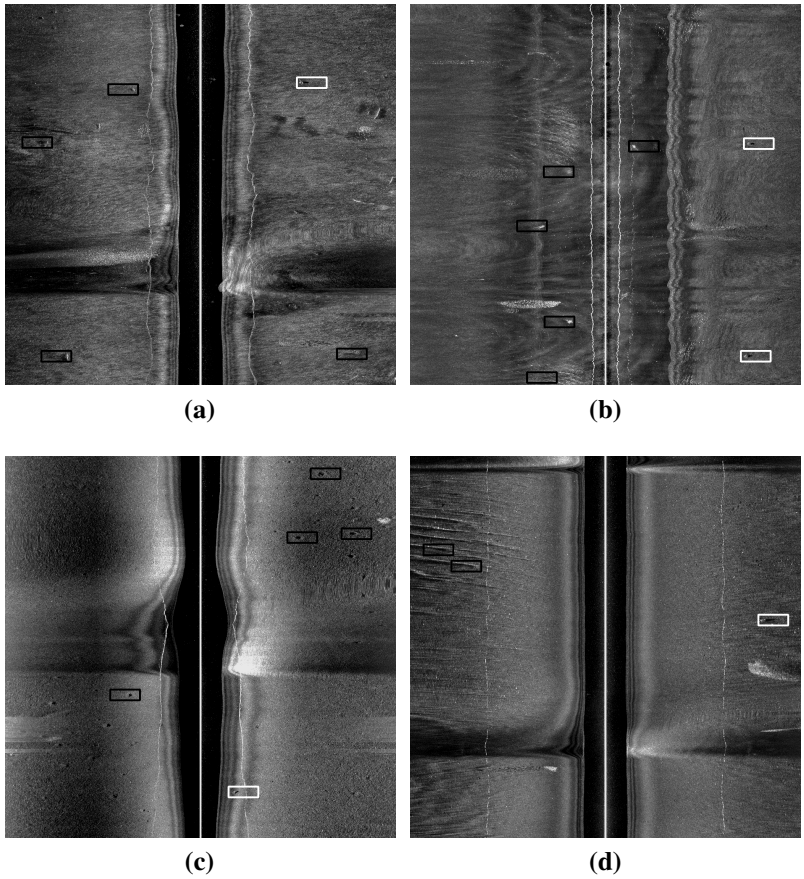


Figure 5.9. Example sonar images including different clutter and sea floor bottoms. Images (a), (c), and (d) contain results from the vehicle making a 180 degree turn. Image (a) shows some clutter objects, image (b) shows a poor quality image with highlight clutter, and image (c) shows small holes. Image (d) shows a sand ripple bottom type.

Most prominently, this data was collected via a star pattern over a mine object, causing most images to have at least one turn. These turns are evident in Figure 5.9 images (a), (c), and (d). Image (a) shows some clutter in the form of foreign objects or rocks on the sea floor. Image (b) shows a low quality image with highlight clutter

spots. Image (c) shows clutter in the form of small holes in the sea floor, which look like shadows. Finally, image (d) shows some sand ripples on the sea floor. All of these slight complexities and clutters are common in this dataset.

Prior to training a classifier on the training dataset, the positive and negative windows must be calculated based on the labeled location of the MLOs. For the local feature types we are using, the positive examples must be aligned for consistency. All windows are treated as if they are on the left side of the image to keep shadow location consistent. We choose to right justify and vertically center the MLO in the window with a two pixel buffer. Figure 5.10 shows three examples of the alignment of positive examples. Note that the mine highlight is consistently located even though the shadows are of varying sizes. The negative window calculated during preprocessing includes the half of the image that does not contain the labeled MLO. The boosting algorithm selects negative example windows of the prescribed window size during training.

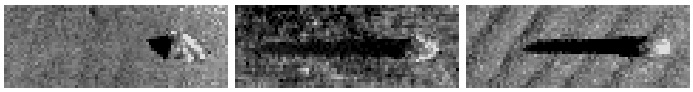


Figure 5.10. Examples of the alignment of positive examples. These positive MLO examples have varying shadow lengths, but an approximately consistent highlight location within the window.

5.4 Experimental Results

The experimental results are separated in to four subsections. The first subsection describes the single-feature selection for each of the six features, which is used to determine the feature that is most capable for this application as well as the best single-feature classifier for a baseline. The next two subsections will build on this knowledge to explore variations of a multi-feature selection, focusing on double-feature selection and triple-feature selection respectively. The final subsection will summarize the experiments

and consider a classifier trained with all six features in the selection pool.

Before looking at the results, we discuss a preprocessing technique intended to find MLOs on the edges of the image. The combination of the fixed window size with the alignment causes targets on the edges of the image to be missed consistently. In order to resolve this we pad the edges of the image with extra data, allowing the search to continue past the original edge. Figure 5.11 shows the upper left corner of an image before and after this padding process. In the left image there is a false negative for a mine on the edge of the image.

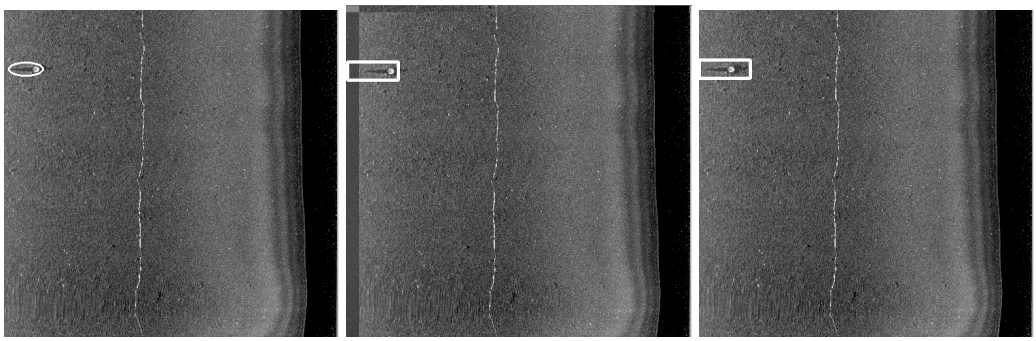


Figure 5.11. Example of a MLO along the image edge. The left image is the default search, which does not find the MLO and is shown by an ellipse. The padded image is shown in the middle, with the target detection shown by a rectangle. The right image shows the detection within the bounds of the original image via a rectangle.

We pad the left and right edges with 20 extra pixels and the top and bottom edges with 10 extra pixels. The pixels come from the opposite edge of the image, in order to get approximate pixel intensity consistency. For example, in order to pad the left edge, an extra 20 pixels in width are copied from the right edge, mirrored along the horizontal axis, and then placed in the extra region on the left edge. The middle image in Figure 5.11 shows the padded image with the positive search result. In the right image of Figure 5.11 the mine on the edge is found within the padded image, then the positive window is moved to the original window's edge in order to be valid in terms of the original image.

This process allows us to use our same quality classifier to find mines that are too close to the edge for our fixed window sliding window paradigm.

In order to produce all of the receiver operating characteristic (ROC) curves in the following subsections, we consider variations of the *minimum neighbors* parameter of our classifier described in Section 5.1. We consider this parameter from zero to 50 *minimum neighbors*.

5.4.1 Single-Feature Results

In order to identify a baseline for comparison, we train a single-feature classifier for each of the six presented features. The classifiers were trained using the GentleBoost algorithm for feature selection with the standard parameters as defined in Section 5.1 and using the training and testing datasets described in Section 5.3.

Table 5.1. Shows the training metrics for the classifiers trained on each of the six single-feature pools. The pool size is the number of features in the feature selection pool, while the stages and features are the respective numbers of each chosen by the GentleBoost single-feature selection algorithm.

Classifier	Pool Size	Stages	Features
HAAR	2,543,145	9	61
HOG	4,032	14	247
LBP	138,645	13	100
SURF	2,820	14	1,600
SHADOW	748	14	1,312
SONAR	646,866	13	158

Table 5.1 shows the training attributes for the six single-feature classifiers. The pool size is the number of variations of the single feature in the selection pool as defined in Section 5.2. The table also shows the number of stages and the number of total features selected by the GentleBoost single-feature selection algorithm.

Notice that three of these classifiers have fourteen stages, which is the maximum number of stages based on our parameters. This implies that further iterations could produce better results. For completeness, we trained each of the three with a twenty stage maximum to see if we would have improved results, and both *SURF* and *SHADOW* classifiers continued through all twenty stages without a substantial benefit to capability. The *HOG* classifier training exited early after seventeen stages due to lack of available training data, which means it used all of the available positive examples to select the features up to that point and could not continue. Training another *HOG* classifier with a seventeen stage maximum produces a classifier with improvements over the 14 stage version. For consistency in boosting parameters between classifiers, we only present the fourteen stage classifiers in this chapter.

The ROC curves for the three best single-feature classifiers are shown in Figure 5.12. Only the first three are shown because the other three do so poorly that they would not fit on a reasonable scale. This means that only the three shown have the flexibility to create a strong classifier on their own. This does not, however, mean that the SURF, shadow, and sonar weak features cannot provide a benefit in a multi-feature approach, as we will show.

On the vertical axis there is the true positive rate (TPR), which expresses how many of the potential MLOs the classifier finds in decimal percentage form. On the horizontal axis we have the number of false positives (FP) per square kilometer (km^2) of sea floor coverage. In the figure there are related point triplets with approximately similar FP per km^2 for comparison between the three classifiers. The first triplet is marked by triangles at a FP per km^2 of approximately 1.25 where the LBP classifier has the best TPR at 55.05%. The next two pairs, marked by squares at approximately 3 and by circles at approximately 8.1 FP per km^2 respectively, both show the *HAAR* classifier with the best TPR.

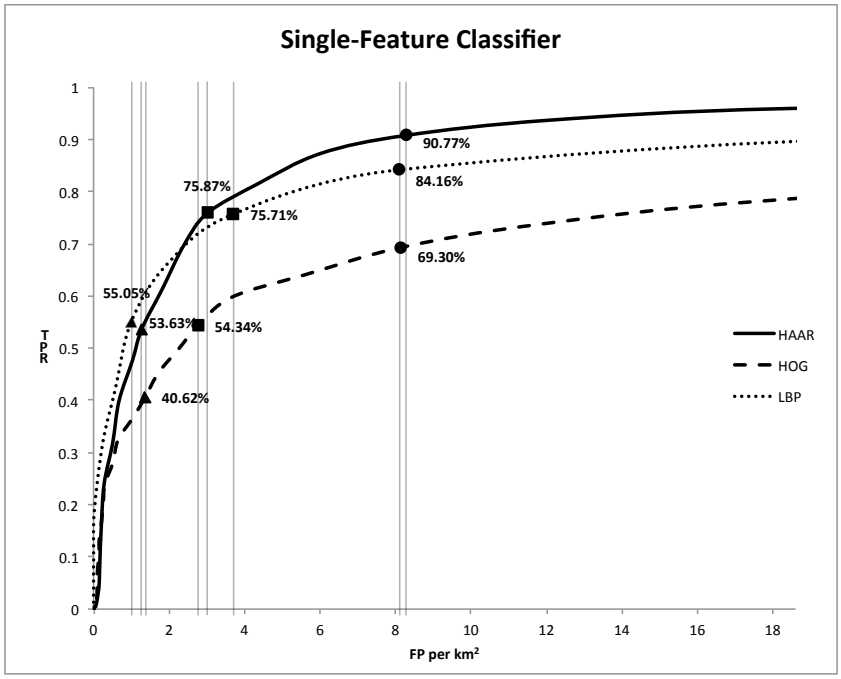


Figure 5.12. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives (FP) per km^2. Includes the ROC curves of the three best single-feature classifiers, while the other three do not fit on this scale due to poor performance. This figure shows that the *HAAR* classifier performs best and therefore should be the baseline for these experiments.

A strong TPR capability is important for the detection of mine-like objects since missing targets can have strong implications. An increase in false positives can cost time but a reduction in TPR means a danger to equipment and personnel. Based on the fact that the *HAAR* classifier performs best for TPR above 75%, we select this curve as our baseline for comparison to our multi-feature classifiers. Also, the results shown here lead us to run experiments on the five double- and ten triple-feature classifier combinations including the Haar-like feature.

5.4.2 Double-Feature Results

There are five combinations of double-feature pools that we can create with the Haar-like feature always included based on its proven capability in Subsection 5.4.1. In

this subsection we introduce these classifiers and compare them to the baseline *HAAR* classifier. All of these were trained under the same parameters and using the same datasets as the single-feature classifiers.

Table 5.2. Shows the training metrics for the classifiers trained on each of the five double-feature pools. The pool size is the number of total features in the feature selection pool, while the stages and features are the respective numbers of each chosen by the GentleBoost double-feature selection algorithm. The subrows of features show the number of Haar-like and other features chosen for the classifier, as well as the total. The other feature is listed in the classifier name in addition to the Haar-like feature.

Classifier	Pool Size	Stages	Features		
			HAAR	Other	Total
HAAR	2,543,145	9	61	*n/a*	61
HAAR + HOG	2,547,177	9	48	2	50
HAAR + LBP	2,681,790	8	47	3	50
HAAR + SURF	2,545,965	8	48	3	51
HAAR + SHADOW	2,543,893	9	53	6	59
HAAR + SONAR	3,190,011	8	50	1	51

Table 5.2 shows the attributes of the five double-feature classifiers including the size of the pool, the number of both stages and features in the classifier. In addition, the features are split in to feature type providing an idea of impact of the extra feature in the classifier. The other feature column corresponds to the non Haar-like feature in the classifier name. The table also includes the attributes of the baseline *HAAR* classifier.

Notice that the number of stages in the classifiers stay nearly the same, but that the number of features chosen reduces for all of the combinations. For each double-feature classifier, the GentleBoost feature selection algorithm selects at least one variation of the alternative feature. This does not necessarily translate to an improvement in capability in the context of the low false positive with high true positive target area for the application.

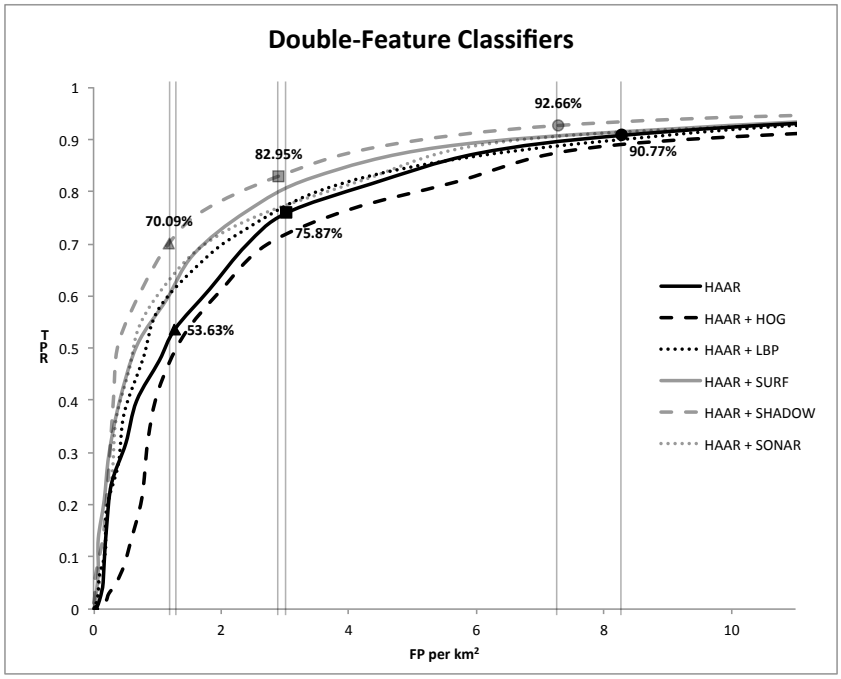


Figure 5.13. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives (FP) per km^2. Includes the ROC curves of the five double-feature classifiers and the baseline *HAAR* classifier. This figure shows that the additional feature type per classifier is able to provide performance improvements for four of the five, with *HAAR + SHADOW* performing the best.

Figure 5.13 provides the ROC curves for each of the five double-feature classifiers in addition to the *HAAR* classifier. Again the vertical axis is TPR while the horizontal axis is FP per km^2 of sea floor coverage. The primary element to note in this figure is that four of the five double-feature classifiers outperform the *HAAR* classifier, with the *HAAR + SHADOW* classifier performing the best.

The *HAAR + HOG* classifier performs worse than the *HAAR* classifier, which might not be intuitive. As each classifier is being trained, the Haar-like feature is selected early for both because it is better at dividing the training data in to positives and negatives. The first 18 features are selected exactly the same between the two classifiers. The next feature causes a divergence as the *HAAR + HOG* training selects a HOG feature.

From this point forward the selections will be very different based on that decision point. This is the same for all of the classifiers, with similarities in the number of Haar-like features chosen before an additional feature is selected. For the *HAAR + HOG* classifier it happens that this divergence leads to a less capable classifier when processing the testing dataset.

The largest improvement in the double-feature classifiers is marked by the triangle point pair at approximately 1.2 FP per km^2, where the *HAAR + SHADOW* classifier provides a TPR improvement of 16.46%. The next comparison is marked by the square point pair at approximately 3 FP per km^2, where the *HAAR + SHADOW* classifier improves the TPR by 7.08% moving the performance above 80%. The last improvement shown has a wider range for FP per km^2 but shows an improvement on both axes. Marked by the circle point pair at approximately 7.75 FP per km^2, the *HAAR + SHADOW* classifier improves TPR by 1.89%.

Considering the capabilities of these double-feature classifiers, we see that the addition of certain features to a multi-feature training framework can lead to substantial benefits under the same parameters and data for training and testing. Notably the combination of a very powerful local feature with a proven model type shadow feature performs very well. We will show in Subsection 5.4.3 that adding a third feature option to the multi-feature selection pool can also lead to further benefits.

5.4.3 Triple-Feature Results

The triple-feature selection use of this framework now considers variations of feature pools containing three features and including the Haar-like feature. The results presented in Figure 5.13 suggest that the pools containing the Haar-like feature along with the SURF feature or the shadow feature should be most likely to produce the best results. We consider all ten combinations of triple-feature classifiers that include the

Haar-like feature with the expectation that the classifiers also containing the shadow feature will perform well.

Table 5.3. Shows the training metrics for the classifiers trained on each of the ten triple-feature pools. The pool size is the number of total features in the feature selection pool, while the stages and features are the respective numbers of each chosen by the GentleBoost triple-feature selection algorithm. The subrows of features show the number of Haar-like features chosen versus the number the other features chosen for the classifier, as well as the total. The other features correspond to the features in the classifier name in order.

Classifier	Pool Size	Stages	Features			
			HAAR	Other 1	Other 2	Total
HAAR	2,543,145	9	61	*n/a*	*n/a*	61
HAAR + HOG + LBP	2,685,822	9	47	2	8	57
HAAR + HOG + SURF	2,549,997	8	49	1	3	53
HAAR + HOG + SHADOW	2,547,925	9	50	2	5	57
HAAR + HOG + SONAR	3,194,043	9	48	2	0	50
HAAR + LBP + SURF	2,684,610	8	44	2	3	49
HAAR + LBP + SHADOW	2,682,538	9	45	3	5	53
HAAR + LBP + SONAR	3,328,656	8	47	3	0	50
HAAR + SURF + SHADOW	2,546,713	9	45	2	7	54
HAAR + SURF + SONAR	3,192,831	8	48	3	0	51
HAAR + SHADOW + SONAR	3,190,759	9	49	6	1	56

Table 5.3 shows the attributes of the ten triple-feature classifiers including the size of the pool, the number of both stages and features in the classifier. In addition, the features are split in to feature type to provide an idea of impact of the extra two features in the classifier. The other feature columns correspond to the extra features in the classifier name in order. The table also includes the attributes of the baseline *HAAR* classifier.

One important note is that all of the classifiers have either eight or nine stages consistent with the double-feature classifiers and the *HAAR* classifier. Also note that in some of the triple-feature classifiers containing the sonar feature, the feature selection

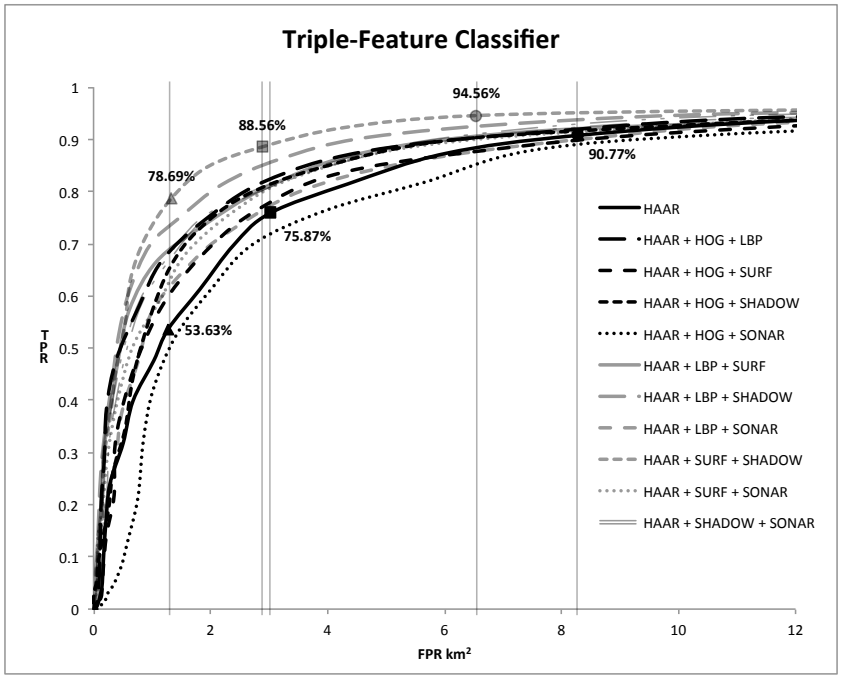


Figure 5.14. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives (FP) per km^2. Includes the ROC curves of the ten triple-feature classifiers and the baseline *HAAR* classifier. This figure shows that the additional two features per classifier is able to provide performance improvements for nine of the ten, with *HAAR + SURF + SHADOW* performing the best.

algorithm did not select any sonar features thus creating a classifier equivalent to a double-feature classifier. For example the *HAAR + HOG + SONAR* classifier ends up being the same as the *HAAR + HOG* classifier since no sonar features were selected. Thus the sonar feature is not adding any additional benefit for three of the triple-feature classifiers tested.

Figure 5.14 provides the ROC curves for each of the ten triple-feature classifiers in addition to the *HAAR* classifier. Again the vertical axis is TPR while the horizontal axis is FP per km^2 of sea floor coverage. Notice that only one classifier, the *HAAR + HOG + SONAR*, performs worse than the baseline *HAAR* classifier. This is not surprising since we have shown that the sonar feature added no benefit to this classifier and the

HAAR + *HOG* classifier performed poorly in Section 5.4.2.

The largest improvement is marked by the triangle point pair at approximately 1.3 FP per km^2, where the *HAAR* + *SURF* + *SHADOW* classifier provides a TPR improvement of 25.06%. The next comparison is marked by the square point pair at approximately 3 FP per km^2, where the *HAAR* + *SURF* + *SHADOW* classifier improves the TPR by 12.69%. The last improvement, marked by the circle point pair, again has a wider range for FP per km^2 but shows an improvement on both axes. At approximately 7.75 FP per km^2 the *HAAR* + *SURF* + *SHADOW* classifier improves TPR by 3.79%.

The results shown in this section emphasize the possibilities provided by the multi-feature selection framework. The *HAAR* + *SURF* + *SHADOW* classifier achieves its substantial improvement in performance using the same training and testing dataset under the same training parameters other than the pool of features.

5.4.4 Multi-Feature Results

The framework presented in this research can train classifiers with more than three feature types in the pool of features. In addition to the systematic research of the single, double, and triple-feature classifiers we train the classifier with all six features in the pool of features. The resulting *ALL* classifier has nine stages with 55 features including 42 Haar-like, 5 LBP, 2 SURF, and 6 shadow features. The GentleBoost algorithm did not select any HOG or sonar features from the pool. The *ALL* classifier outperforms the single *HAAR* classifier but does not improve over the *HAAR* + *SURF* + *SHADOW* classifier.

Figure 5.15 shows the best single, double, and triple-feature classifier from this research and includes the *ALL* classifier for comparison. The figure shows that the *HAAR* + *SURF* + *SHADOW* triple-feature classifier outperforms the other classifiers on this data, achieving an improvement of 12.69% in TPR at approximately 3 FP per km^2 and 3.79%

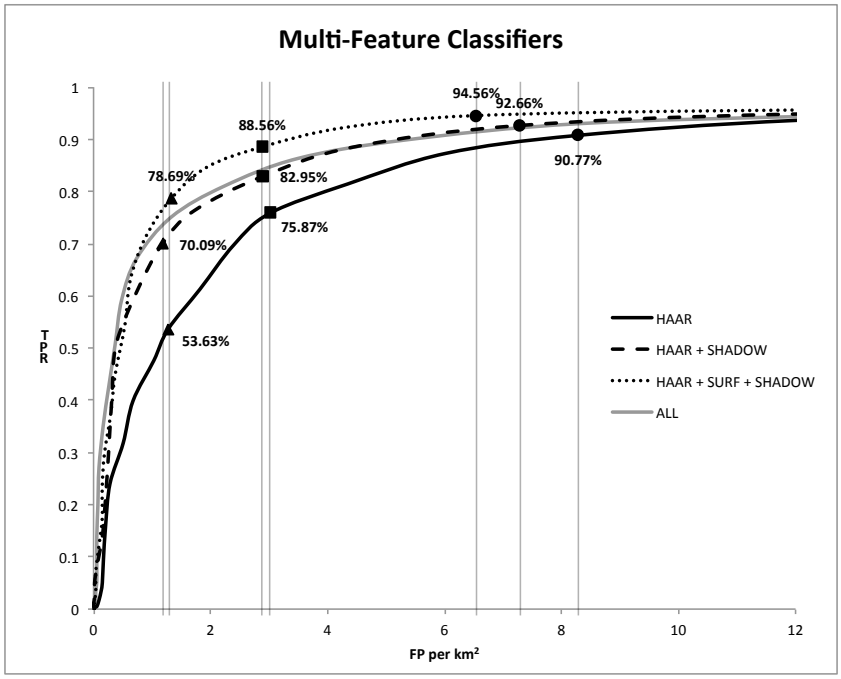


Figure 5.15. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives (FP) per km^2. Includes the ROC curves of the best classifiers from the single, double, and triple-feature selection experiments as well as the *ALL* classifier using all six features in the training pool. This figure shows that the additional features in the training pool lead to classifiers with substantial performance improvements, but that more feature types in the pool does not necessarily create the optimal classifier.

in TPR with reduction in FP per km^2 by 1.74 to 6.54 over the *HAAR* classifier. While showing the improvements when the feature pool includes additional feature types, this figure also highlights that more features do not necessarily lead to improved performance. The feature types perform well together when they are complimentary and focus on characteristics that are independent from each other.

5.5 Conclusion

This chapter introduces a multi-feature selection framework for the detection of mine-like objects in side scan sonar imagery. We explain in depth the GentleBoost algorithm and its evolution, describing its potential for multiple feature types in the pool

of features. During these experiments we build on the work of Sawas and Petillot [54] and Hollesen et al. [27] showing the results of other prominent local type features with the GentleBoost single-feature selection. We consider six different features under under single and multi-feature selection frameworks to compare the results.

The obvious extension of this work is to further apply this multi-feature selection framework. There are two avenues for this, one being applying the framework to other applications and the other is to add new features to improve this application. The framework is structured such that new features can easily be added and any dataset as well as any group of features can be used for training and classification.

In addition to these contributions, we also present the optimal combination of feature types selected by the multi-feature selection algorithm. The *HAAR + SURF + SHADOW* classifier is able to achieve 88.56% TPR with only 2.88 FP per km^2 or 94.56% TPR with 6.54 FP per km^2. This is a substantial capability improvement over the other classifier combinations under the same GentleBoost parameters using the same datasets.

This chapter, in full, has been submitted for publication of the material as it may appear in Ocean Engineering, Barngrover, Chris; Ryan, Brett, Belongie, Serge; Kastner, Ryan, 2014. There are small changes in format and phrasing as a chapter within this larger paper. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Tiered Classifiers

The goal of the tiered classifiers experiment is to reduce some of the false positives in the standard processing routine. The reasoning is that if a first classifier is able to create a secondary training set that is more focused on the harder variations between true positives and false positives, then the classifier will have less false positives. Section 6.1 will explain the concept in more detail while Section 6.2 will present the experimental results.

6.1 The Tier Concept

In order to explain the concept of training a classifier in stages, we will review the standard way that a classifier is trained. The classifier here is really separate from the process and can use any training scheme with any features. Figure 6.1 shows a visual representation of the standard flow for training and testing a classifier. To begin, the set of all data is divided in to two separate sets for training and testing. In these experiments the training dataset contains 975 images while the testing dataset contains 920 images. The training step is very simple, the algorithm trains on the training set and produces a classifier. Then in the testing set, the classifier is used to process the testing set and this produces a set of positive windows or locations in certain images.

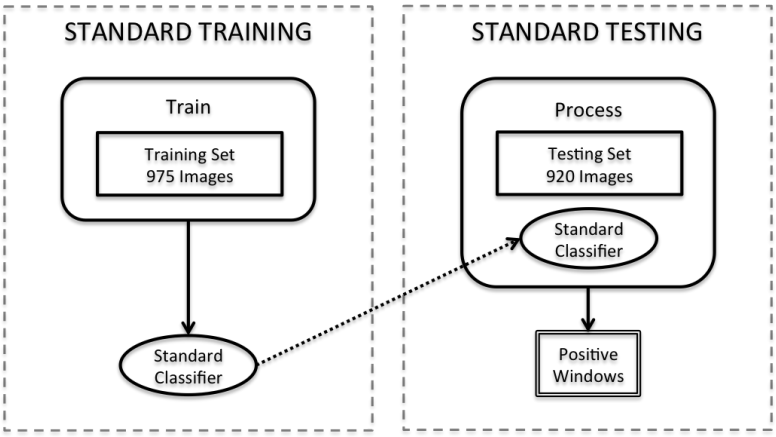


Figure 6.1. A visualization of the standard flow for training and testing a classifier. The training algorithm is applied to the testing set to produce a classifier. This classifier is applied to the testing set in the processing stage to produce positive windows.

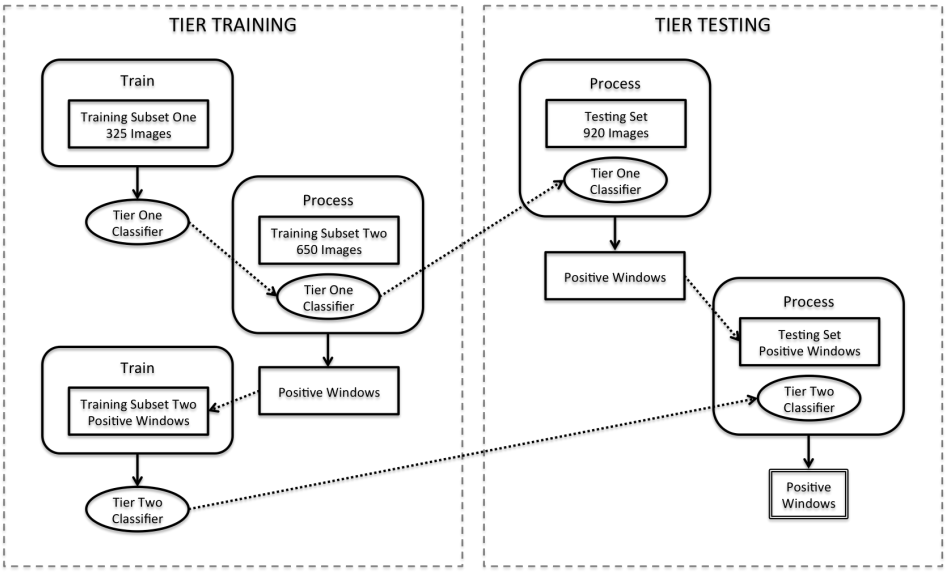


Figure 6.2. A visualization of the tier flow for training and testing a classifier group. The training algorithm is applied to training set one to produce the tier one classifier. This classifier is used to process training set two and produce a set of windows. These windows become the training set for the training algorithm, which produces the tier two classifier. The testing stage uses the tier one classifier to process the testing set and the windows output are then processed with tier two classifier to create the final set of positive windows.

The tier version has the same two stages of training and process, but each of them are slightly more complicated. Again the classifier training scheme is separate from the tier concept. Figure 6.2 shows a visualization of the training and testing flows for the tier process. The training set from the standard process is split into two subsets for the tier process. In these experiments we have 325 images in training subset one and 650 images in training subset two. The same testing set with 920 images is used. The training step starts by applying the training algorithm to training subset one to produce the tier one classifier. The tier one classifier is used to process the training subset two, which produces a set of positive windows. Then the training algorithm is applied to these windows in order to create the tier two classifier. The testing step of the process begins by applying the tier one classifier to the testing dataset, producing positive windows, which are processed by the tier two classifier. The output is a set of positive windows or locations in certain images in the same format as the standard process.

The premise of the tier training is that the first tier will isolate the regions of the image that are more difficult to classify and thus create a more accurate second tier classifier. It is important to note that both of the classifiers can be used individually as standard classifiers. Also, the tier two classifier could be used with another separate first tier classifier for the testing stage. Some of these variations are considered in the following section.

6.2 Experimental Results

The main focus of this research has been using the Haar-like feature with boosting to create a cascade classifier. For this reason, the tier experiments continue to use this training setup in order to compare to a baseline of capability. The same training and testing datasets are used as were used for training and testing the standard classifier. The difference is that the training set is divided in to two subsets for the tier process.

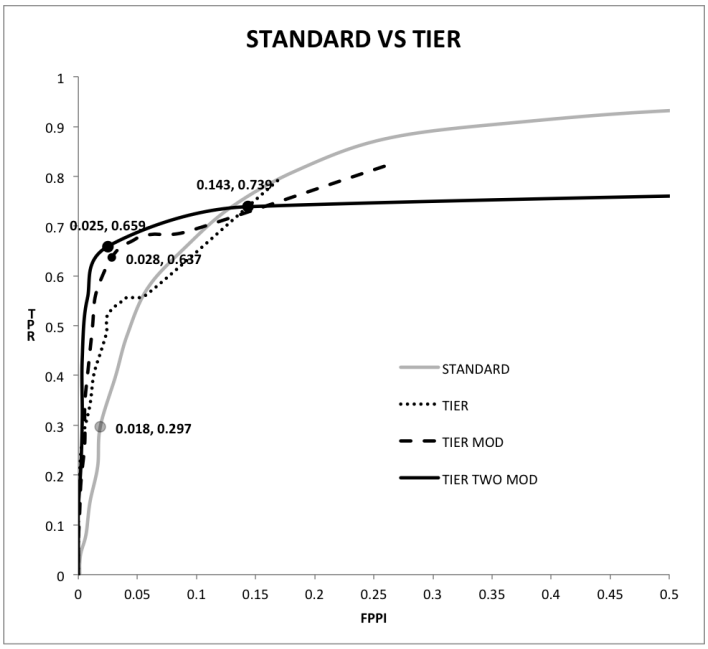


Figure 6.3. The receiver operatinging characteristic (ROC) curve where vertical axis is true positive rate (TPR) and the horizontal axis is false positives per image (FPPI). The *STANDARD* curve is created by processing with the standard classifier. The *TIER* curve is formed by processing with the tier one and tier two classifiers. The *TIER MOD* curve is similar to *TIER*, except the boosting settings are altered based on the nature of the tier concept. The *TIER TWO MOD* is the created using just the tier two classifier from the modified tier training process used to create *TIER MOD*.

The capability of the classifiers are presented as receiver operating characteristic (ROC) curves as normal. Figure 6.3 compares the standard classifier to variations of the tier classifier. The *STANDARD* curve is from the classifier trained in the standard manner, while *TIER* curve is from the classifier trained with the tier process described in this chapter. The *TIER MOD* curve uses a classifier trained with the tier process, but with non-default settings for the boosting algorithm. The *TIER TWO MOD* curve is created using just the tier two classifier from the same *TIER MOD* training scheme in a standard testing setup.

The boosting modifications for *TIER MOD* alter the hit rate and false alarm rate. Specifically the hit rate is set to 0.999 and the false alarm rate is set to 0.8 as compared to

the defaults of 0.995 and 0.5 respectively. The hit rate is the number of targets correctly classified as positive out of the total number of positive windows considered. The false alarm rate is the number of targets incorrectly classified as positive out of the total number of negative windows considered.

There are three interesting points highlighted in Figure 6.3. First there are the left two points which show the relative improvement in TPR capability between *STANDARD* and *TIER TWO MOD*. For an approximate FPPI of 0.02 there is a 36.2% improvement in the TPR when using the *TIER TWO MOD* classifier. Also note that the two curves cross over at the other highlighted point.

Figure 6.4 considers another variation of this tier concept. In this scenario a

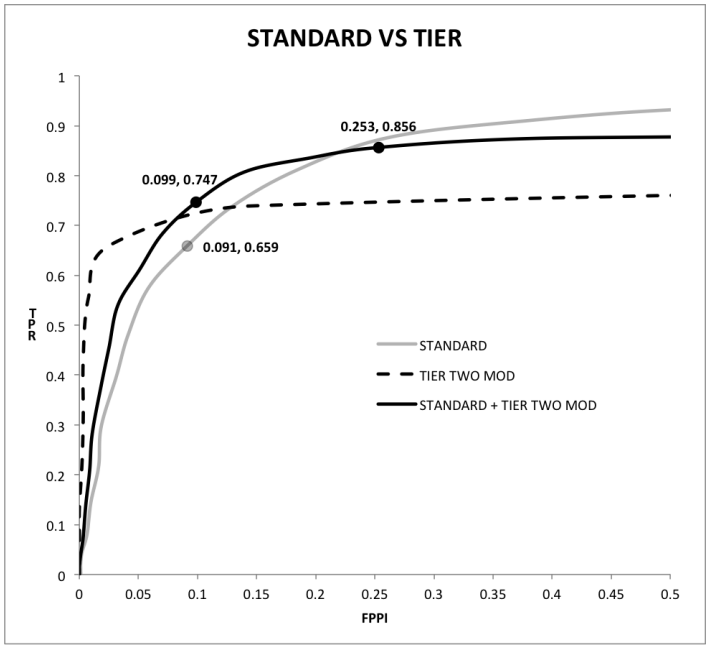


Figure 6.4. The receiver operating characteristic (ROC) curve where vertical axis is true positive rate (TPR) and the horizontal axis is false positives per image (FPPI). The *STANDARD* curve is created by processing with the standard classifier. The *TIER TWO MOD* is the created using just the tier two classifier from the modified tier training process as in Figure 6.3. The *STANDARD + TIER TWO MOD* curve is formed using the *STANDARD* classifier as the tier one classifier and the *TIER TWO MOD* classifier as the tier two classifier in tiered processing.

truncated version of the classifier trained in the standard manner is used as the tier one classifier and the tier two classifier from tier training is used as the tier two classifier. The truncated version means that the window must only pass the first four stages of the classifier to be marked as positive. The result is shown in Figure 6.4 as *STANDARD + TIER TWO MOD*. The other two curves are the same described in Figure 6.3.

There are also three highlighted points in this figure, showing improvements and crossover point of the new curve compared to the *STANDARD* curve. The first two points show the improvement of the *STANDARD + TIER TWO MOD* variation of tier classifier over *STANDARD* and only a slight improvement over *TIER TWO MOD*. The points show a TPR increase of 8.8% at an approximate FPPI of 0.095.

6.3 Conclusion

The results of tier training show promise at improving a classifiers capability in the low false positives domain. This concept of tier training and tier testing was only a small portion of this research and did not encompass the same level of related works investigation or as in depth of experimentation. Further experimentation is necessary to determine if this concept has merit. This chapter is meant to describe the effort so that it might be carried on in the future.

Chapter 7

Multiple Instance Learning

The traditional form of supervised learning is to have a single positive example paired with its positive label and to have a single negative example paired with its negative label. In all other chapters we use a traditional learning paradigm using boosting with positive mine windows and negative background windows for training. In this chapter we experiment with a different training concept called multiple instance learning (MIL). Specifically we will us the combination of this MIL training concept with the boosting algorithm, using the MILBoost algorithm for training.

This chapter will discuss the concept of MIL and how it compares to the more traditional training structure. The MILBoost algorithm will be explained in depth and some of the available frameworks for the algorithm will be outlined. Most importantly this chapter will present the experiment we designed and the results before discussing the future opportunities for MILBoost and this particular application.

7.1 Multiple Versus Single Instance Learning

Single instance learning uses training data in the form of a pair (x_i, y_i) where x_i is the example, a window from a sonar image in our case, and y_i is the label, either target or non-target. Since the examples are paired with a label the training considers the one instance as an example of positive or negative. The paradigm shift in MIL is that the

same label, y_i is paired with a "bag" of examples $X_i = \{x_{i1}, x_{i2}, x_{i3}, ..., x_{im}\}$. A given bag with a positive target label means that at least one image in the bag is a positive, while a bag with a negative non-target label is exclusively composed of negative examples [14].

In the context of images, the MIL concept is ideal for scenarios where the position or pose of the target is not known or the estimation of the location has some error. This allows the algorithm to place in a bag all of the image windows around the estimated location of the target and label this bag positive. Similarly, a bag of negatives can be created by excluding the estimated location of the target.

For the detection of mine like objects in side scan sonar, the location can be inaccurate, the size of the shape is very small leading to training errors, and the shadow length can vary greatly causing the relative location of the mine in the window to be important. With the single instance learning we describe the right justification of the mine target in the window for consistency. The use of MIL allows us to skip this decision and place a group of various window locations around the target in to a single positive bag.

7.2 MILBoost

The next step is to alter the boosting algorithm to use this MIL paradigm [68]. We focus on the Noisy-OR implementation of the MILBoost algorithm. In standard boosting a linear combination of weak learners classifies each example, but in MILBoost the the examples are in bags, not individual. Thus the examples are double indexed with an i indexing the bag and a j indexing the index within the bag.

The score for a given example y_{ij} is represented by the weighted sum of weak classifiers, $C(x_{ij}) = \sum_t \lambda_t c^t(x_{ij})$, where $c^t(x_{ij})$ is the weak classifier score function. A given round of boosting is searching for a classifier that maximizes $\sum_{ij} c(x_{ij}) w_{ij}$ where $c(x_{ij})$ is again the score function for the weak classifier and w_{ij} is the weight given to the

example.

The MILBoost algorithm ends up with a weighting on each bag, W_{bag} and a weighting on an individual example, $W_{instance}$. For negative examples the $W_{bag} = -1$. This means that, as with standard boost, the negative examples are all equally negative, and therefore the weight of a negative is the same with MILBoost as it would be with standard boosting. For positive bags, the examples with higher scores get higher weights and in turn dominate the learning in the future. Thus the examples in the positive bag are not created equal, as they would be in a standard boosting algorithm.

There are a few open source frameworks available to us, but the primary problem is that they do not account for large datasets. Our data quickly fills memory and will not work. The result is that we implement our own version by combining some of our OpenCV based code with the libraries used in the MILTrack application [5]. We perform some of the calculations in batches in order to avoid the memory issues and avoid completely rewriting existing MILBoost code.

7.3 Experimental Results

We apply our version of the MILBoost algorithm to our application of mine-like objects in side scan sonar. In order to compare the results to the standard Haar-like feature cascade, we use the same training and testing datasets.

The first step in the training is creating one positive bag and one negative bag for each image. The negative bags are easy, because they are simply random background windows from around the image. The positive bag is filled with various views of the positive region. We start with the same window over the mine as we would with standard boosting, but then we slide the window pixel by pixel along each axis until we have added all variations out to five pixels in all directions. Figure 7.1 shows an example of some of the positive and negative windows added to positive and negative bags.

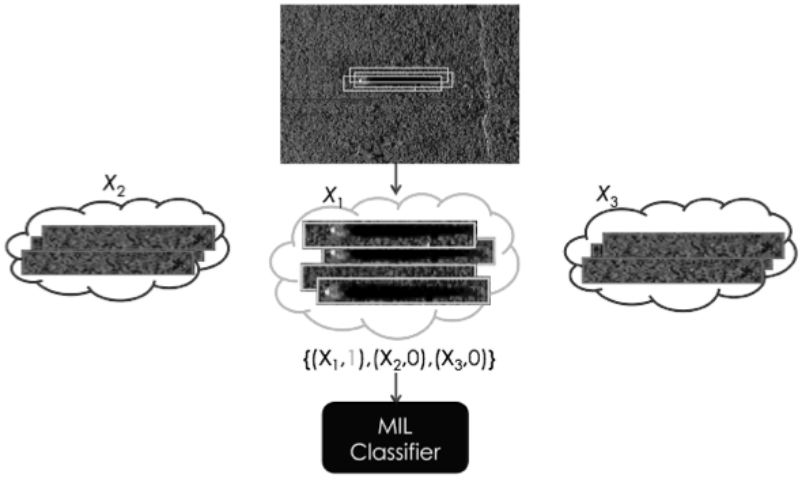


Figure 7.1. Visualization of the MIL bagging and training process. Each negative bag contains various negative regions. Each positive bag contains pixel varied windows around the positive location. The bags are trained with a variation of standard boosting to create a cascade classifier.

This training process is very inefficient and is only meant to determine the capability of a resulting classifier rather than designed for speed. Because of this, the classifier was only trained for three stages. The resulting classifier is then used to process the testing dataset. The receiver operating characteristic (ROC) curve of the MILBoost classifier is plotted next to the ROC curve of the standard boosting classifier in Figure 7.2. The vertical axis shows the true positive rate (TPR) while the horizontal axis shows the false positives per image (FPPI).

The obvious point of interest in this figure is at an FPPI of 1.1 and TPR just over 90%. For lower than 1.1 FPPI, the standard classifier performs better and for a higher FPPI the MILBoost classifier performs better. This means that there could be a benefit to MILBoost when the number of true positives is the main concern, or if the classifier is the first stage of a pipeline of processing. Furthermore, the MILBoost classifier was only trained to three stages and one would expect an improvement with more training.

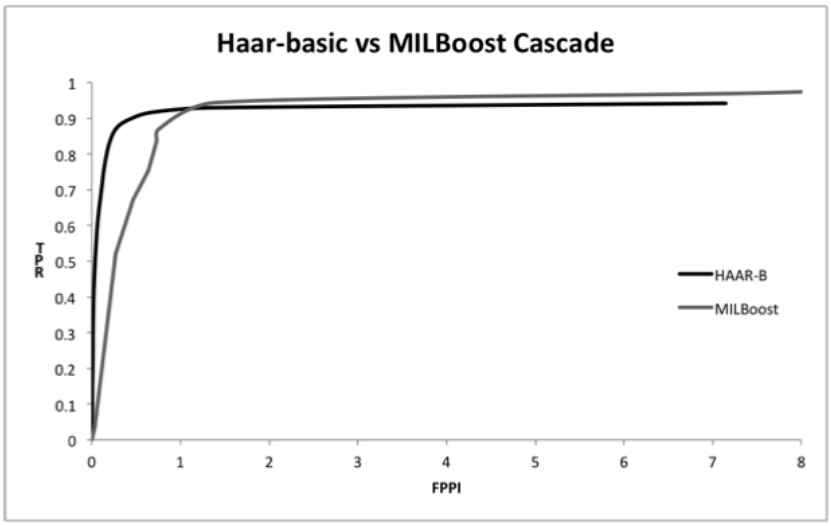


Figure 7.2. The ROC curves for the MILBoost and the standard boosting (HAAR-B) classifiers on the same testing dataset. The vertical axis is the true positive rate (TPR) and the horizontal axis is false positives per image (FPPI).

7.4 Future Efforts

There are many elements to this line of research for this application that merit further investigation. First and foremost is an efficient version of the algorithm that can take large amounts of data without loading all images into memory at once. This would be a benefit, not just for this application, but for the computer vision community as a whole. The OpenCV framework is an obvious place to add this functionality and would be a logical extension of this effort.

In addition, the current three stage classifier we present in this chapter is only a glimpse of what the classifier might look like. Whether via the current inefficient implementation or with a yet to be implemented version, the classifier needs to be trained for more stages to appropriately compare it with the standard boosting classifier or other capabilities.

7.5 Conclusion

We have presented the multiple instance learning (MIL) concept for training and the MILBoost training algorithm. We described our implementation of this algorithm with its shortcomings and the reasons why existing frameworks did not work. Finally we presented our experiment and the results compared to the standard boosting paradigm, which shows despite limited training that MILBoost has potential to improve detection of mine-like objects in sides scan sonar imagery.

Chapter 8

A Brain-Computer Interface (BCI) for the Detection of Mine-Like Objects in Side Scan Sonar Imagery

The detection and classification of mine-like objects (MLOs) in side scan sonar imagery is a problem of grave importance to the safety of our military. The manual approach is still the primary technique for finding and eliminating these objects. Even with the aid of underwater robotics to capture data, the task of processing the imagery is very time consuming. The automation of these tasks would save time and money, but the dynamic underwater environment makes this a difficult task for classifiers, with the human operators still leading in performance.

There is an extended history of research on the topic of automated detection of mine-like objects in sonar imagery. Much of the earlier research uses a model based approach with knowledge of the target, focusing on highlights and shadows created by the strong reflection and occlusion of the protruding MLO from the sea floor [38, 49, 33]. In some research the model for various range regions is used as a matched filter, convolving the filter with the image to detect regions of interest [17].

The advancements in the capabilities of sonars and the autonomous underwater vehicles utilizing them has led to research using machine learning techniques and well

known computer vision features. These features do not have a concept of the target model and instead focus on local descriptors within a window of the image. The machine learning algorithms require large training datasets to optimize a classifier.

One machine learning capability, called AdaBoost, was altered to be a feature selection process, choosing Haar-like features from a pool of feature options [62]. The research paper by Viola and Jones was proposed as a face detector, but it has been applied to many other targets [39, 36, 30]. This concept has also been applied to MLO detection in sonar, but using a variation of boosting called GentleBoost [54].

While computer vision and machine learning approaches to MLO detection have made significant progress over the past decade, the human visual system's ability to recognize objects of interest remains unmatched. Humans can easily and robustly identify objects in a scene, regardless of the scale, lighting, background clutter, etc. Moreover, when an image is flashed quickly, humans are able to ascertain the *gist* of a scene in as little as a few hundred milliseconds [44]. However, when tasked to process large databases of images, computers have the advantage over humans in terms of processing speed, data throughput, and absence of fatigue.

Many variations of brain-computer interfaces (BCIs) have previously been used to tackle the image search problem [25, 51, 52, 7, 31]. One particular BCI system, Cortically-Coupled Computer Vision (C3Vision), synergistically combines the respective advantages of computer vision and human vision. C3Vision starts with a computer vision system that preprocesses large images into smaller *image chips* around potential regions of interest (ROI). This is done using a model-based approach with the assumption that targets are known *a priori*. This framework contains a feature dictionary, containing extracted low-level features, which is used to infer objects using a grammar-based reasoning engine [52]. C3Vision then presents these *image chips* in a rapid serial visual presentation (RSVP) manner, which maximizes the throughput while still being able to decode neural

activity related to detection and recognition as measured by the electroencephalography (EEG). This RSVP component detects neural attention changes rather than behavioral responses, such as button presses.

For example, in one application the C3Vision system performed well on satellite imagery. In this example, intelligence analysts search for targets of interest, such as surface-to-air missile sites or air fields, within large images on the order of several hundred gigapixels in size. The analysts can rapidly view thousands of *image chips* and identify ROIs, which deserve closer inspection. The C3Vision BCI setup showed that the search process could be accelerated without degraded detection performance [51].

8.0.1 Our Approach

The C3Vision BCI setup has performed well in the past on various datasets using a simple computer vision method to prepare *image chips*. And for the application of detecting MLOs in side scan sonar imagery, we have seen that the GentleBoost Haar-like feature selection algorithm works well [54]. We propose creating a BCI system that uses a trained Haar-like feature classifier to create the *image chips* and the RSVP capability of C3Vision for human processing.

Our approach begins by independently considering an implementation of the Haar-like feature selection algorithm to create a classifier and an experimental setup using the C3Vision RSVP capability with human subjects to rank images of interest. Then the primary experiment of this chapter will be to cascade the Haar-like feature classifier with the C3Vision RSVP to show the improvements over the individual capabilities.

We also consider an additional BCI system, which has three stages. The first two stages are the same as the previous approach, starting with the same Haar-like feature classifier with output processed by subjects using the C3Vision RSVP. The final stage is a support vector machine (SVM) classifier trained using a feature vector composed of the

same selected Haar-like features as well as EEG *interest score* features.

There are four major contributions of this chapter:

- The concept of RSVP with EEG systems and human subjects to the application of detecting MLOs in side scan sonar imagery.
- The introduction of a new BCI system using a Haar-like feature classification stage cascaded in to a RSVP human classification stage for the detection of MLOs in side scan sonar imagery.
- A novel SVM classifier whose training feature vector consists of both Haar-like features and EEG *interest score* features.
- An interesting BCI system with the first two stages of computer vision to create the *image chips* and RSVP human processing, followed by the third stage of a novel SVM classifier.

The remainder of this chapter is organized as follows. Section 8.1 introduces the sonar images dataset and the MLO targets. Then we describe the Haar-like feature classifier and its performance on our dataset in Section 8.2. Next, in Section 8.3, we explain the rapid serial visual presentation (RSVP) algorithm and its performance on our dataset. Section 8.4 proposes a BCI setup that cascades the Haar-like feature classifier before the RSVP, showing three experiments using variations of the computer vision classifier. A novel BCI step is presented in Section 8.5 that uses the Haar-like features and the EEG *interest scores* to train a support vector machine (SVM) classifier, which is used as a third stage following the same two stage setup from Section 8.4. Finally we conclude in Section 8.6.

8.1 Dataset

The data used for this chapter was collected using Remote Environmental Monitoring UnitS (REMUS) vehicles in collaboration with the Space and Naval Warfare Systems Center Pacific (SSC-PAC) in San Diego, CA. The REMUS vehicle is equipped with two Marine Sonic side scan sonars operating at a frequency of 900kHz. The missions were all ran at an altitude of four meters producing images with a 30 meter range from each sonar. The combined sonar image used in this research is 1024 by 1000 pixels.

(a) Type 1. **(b)** Type 2.

Figure 8.1. Examples of the inert mine types contained in the datasets used for this experiment. These objects sit on the sea floor protruding up from the bottom. There are ten different Type 1 mines and seven different Type 2 mines included in our data.

There are two different types of mines, shown in Figure 8.1, placed in test fields in San Diego Bay. Multiple passes of the various mine locations, including ten truncated cones (Type 1) and seven stealth wedges (Type 2), produce the many looks included in our data. The dataset consists of 450 sonar images containing 150 target mines.

The data used in this research is relatively easy for this task, but is congruent with the environments currently considered in real world mine clearing situations. MLO detection in the more cluttered sea bottom environments is an important task, but we focus on the easier, very shallow water (VSW) environments for this research. The data does have some complexity, however, including sand ripples, image quality, rock clutter and the prominent vehicle turn regions as shown in Figure 8.2.

This data was collected via a star pattern over a target, causing most images to have at least one turn. These turns are evident in Figure 8.2 images (a), (c), and (d). Image (a) shows rocks on the sea floor, while image (b) shows low quality with highlight clutter spots. Image (c) shows small holes or dimples in the sea floor, which look like shadows. Finally, image (d) shows some sand ripples on the sea floor. All of these slight complexities and clutters are common in this dataset.

The Haar-like feature classifier in the next section and the RSVP classifier in

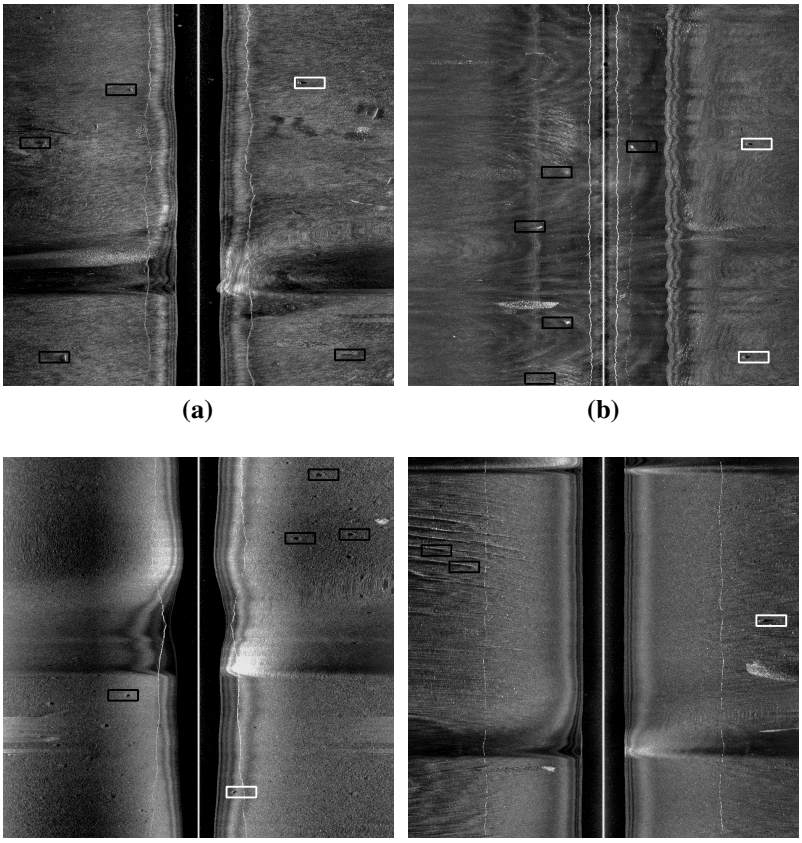


Figure 8.2. Sonar images show the minor complexity of this dataset. Images (a), (c), and (d) contain results from the vehicle making a 180 degree turn. Image (a) shows some rock clutter and other objects, image (b) shows a poor quality image with highlight clutter, and image (c) shows small holes or dimples in the sea floor. Image (d) shows a sand ripple bottom type.

the following section both use all 450 images as a testing set. Section 8.4 also uses this same 450 image dataset as a testing set for the the BCI system including a Haar-like feature classifier cascaded in to a RSVP. Finally, Section 8.5 divides this dataset in to 225 training images to create the new SVM classifier and 225 testing images to test the three stage BCI setup.

8.2 Haar-like Feature Classifier

The Haar-like feature classifier used in this research is produced using the feature selection algorithm proposed by Viola and Jones [62] and later applied to MLOs by Sawas et al. [54]. The classifier is composed of a series of stages, each of which is a classifier in itself, which are cascaded to speed up the processing time. This is a sliding window classifier with a fixed window size and a horizontal step of one pixel and a vertical step of two pixels. Figure 8.3 shows the cascade concept, with each stage composed of multiple features represented by squares, and a positive output from one stage required for entrance in to the next stage.

The algorithm is created using a boosted feature selection process to select the features of each stage and to determine when a stage is complete. The pool of features from which the algorithm selects includes variations of the haar-like feature. Both the feature selection algorithm and the feature itself are described in more detail in the following two subsections. In Subsection 8.2.3 we present the experimental results of this classifier on our dataset.

8.2.1 GentleBoost Feature Selection

The GentleBoost feature selection algorithm combines improvements in boosting algorithms with the Viola and Jones [62] proposal for feature selection to improve the classifier optimization process. In a boosting algorithm, each iteration estimates based on

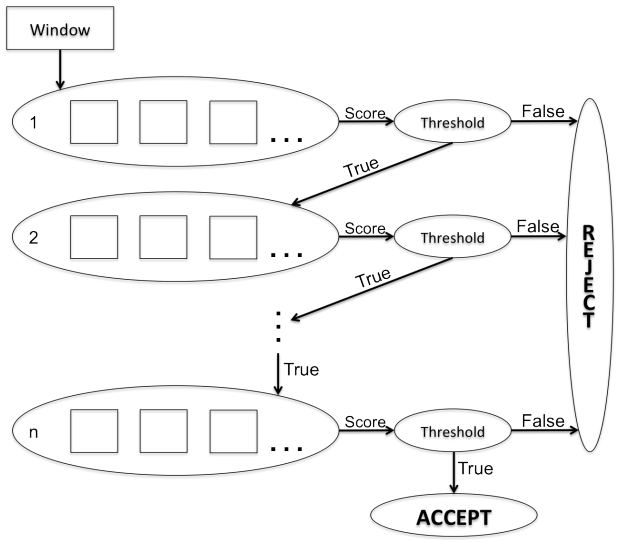


Figure 8.3. The schematic view of a cascaded classifier, with the large ovals to the left being stages and the squares within them being features. Each classifier stage produces a score that is tested against a threshold to determine if the stage is passed. If any stage threshold is not passed then the window is rejected. If all stages are passed then the window is accepted.

the current classification scheme and then updates the classifier based on the error from ground truth. The rule to update the classification scheme is based on the weighting of the training data. This update to the weights is what changes between versions of boosting algorithms. All versions use training data (x_i, y_i) where x_i is the data and $y_i = \{-1, 1\}$ is the label for positive or negative.

The GentleBoost variation of boosting uses adaptive Newton steps to update the classification rule rather than the more volatile log-ratio update of previous Real AdaBoost method [23]. This allows for smaller changes to weight distribution at a given update step compared to the potentially large updates in Real AdaBoost. The classification output is a real value where the sign gives the classification and magnitude gives the confidence. The weight update function is shown in Equation 8.1. Instead of taking the log-ratio of the probabilities given by the classification hypothesis, $h_t(x)$, the

more stable difference is used.

$$w_i \leftarrow w_i \exp[y_i \times (1 - 2h_t(x_i))] \tag{8.1}$$

The feature selection component involves a change to the traditional boosting algorithm, where each iteration restricts its consideration to one feature at a time. The process operates by iteratively selecting new features until the goal true positive rate (TPR) and false positive rate (FPR) are met. The TPR is the number of correctly labeled targets out of the total targets in the training data, while the FPR is the number of incorrectly labeled targets out of the total number of negative windows considered. In the case of our training, the goal TPR is 99.5% and the goal FPR is 50%. The algorithm will continue adding stages to the cascade until a set number stages are created, fourteen in this research, or an acceptance ratio is achieved.

8.2.2 Haar-like Feature

The pool of features from which the GentleBoost feature selection algorithm creates the classifier consists of variations of the Haar-like feature. This feature is based on the haar wavelet and captures changes in pixel intensity between neighboring regions. There are many variations of the the feature but this classifier only considers five basic types, which are visualized in Figure 8.4.

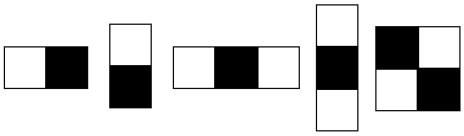


Figure 8.4. The visualization of the five Haar-like features that we use in this chapter. The sum of the pixels in the white rectangle is subtracted from the sum of the pixels in the black rectangle to calculate the Haar-like feature value.

The actual feature is calculated by taking the difference between the sum of pixel intensities in the white regions and the sum of the pixel intensities of the black regions shown in the visualizations. A particular Haar-like feature consists of the feature type, the location within the window being considered, and the size of the rectangles. The pool of features considered by the feature selection algorithm includes all possible variations of these attributes for our fixed window size of 79 by 29 pixels, which amounts to 2,543,145 Haar-like features.

8.2.3 Results

The Haar-like feature classifier we use in this chapter was previously trained by the GentleBoost feature selection algorithm on another dataset collected under the same exact parameters as the data presented in Section 8.1. The training set used contains 975 sonar images, each containing an example mine target. This classifier consists of seven stages and 36 total features. Here we present how this classifier performs on our dataset of 450 sonar images with 150 mine-like targets.

The receiver operating characteristic (ROC) curve representing the performance of this classifier on the dataset is presented in Figure 8.5. The vertical axis shows the true positive rate (TPR), which is the number of targets correctly labeled out of the 150 possible targets in the dataset. The horizontal axis shows the false positives per square kilometer (FP per km^2), which is the number times the algorithm incorrectly labeled a window as a target in terms of a square kilometer of the sea floor. We are able to produce this value based on the known range of the sonar images in this dataset.

The points that make up the curve are created by processing the dataset with the classifier under a varying threshold value. This threshold value is the number of positive window neighbors necessary to mark a window as positive. For example, if the threshold value is zero then any window that is classified as positive by the cascade is output as

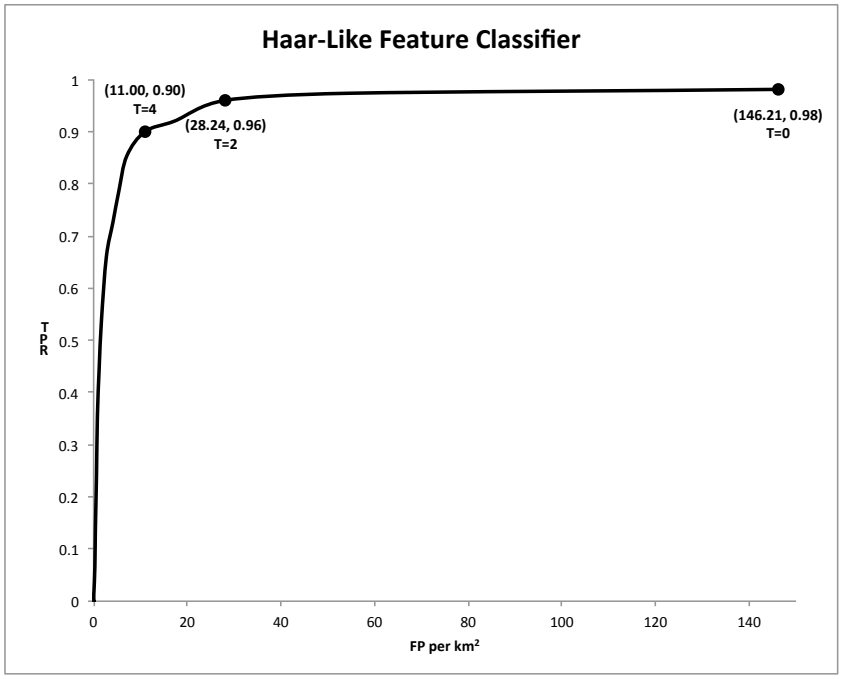


Figure 8.5. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). Shows the receiver operating characteristic (ROC) curves of the Haar-like feature classifier. The three threshold points of interest are highlighted on the curve.

positive. Alternatively, if the threshold is two, then a window that is positively classified must have two other neighboring windows also classified as positive for the output to be positive. In other words, the higher the threshold, the more conservative the classifier is about outputting a positive label for a window. Figure 8.5 highlights three points on the curve where the threshold is set to zero, two, and four respectively.

8.3 Rapid Serial Visual Presentation

The human visual system has the ability to comprehend the *gist* of scenes when presented with images in rapid succession at a rate of five to ten images per second. Using this rapid serial visual presentation (RSVP) paradigm allows for the maximization of image throughput while maintaining the ability to decode electroencephalography

(EEG) signals related to moments of visual interest.

The C3Vision system, used in this research, has an RSVP setup that is designed to distinguish between two brain states: moments of visual interest triggered by positive *image chips* containing something salient versus idle moments when negative *image chips* do not contain anything of particular interest. It is important to note that the system does not decode brain signals based on what exactly the user sees in an image, which would be a very difficult problem to solve, but rather marks when in time a subject detects something of interest. Also, the RSVP system detects neural attention changes rather than behavioral responses, such as button presses.

In order to discriminate between positive and negative examples, an EEG interest classifier must be calibrated for an individual subject, since each subject has a different EEG marker for a moment of visual interest. Therefore, a short calibration session is required for each user of the RSVP setup in C3Vision. During the calibration, sets of known targets and non-targets are presented to the user in RSVP to calibrate the EEG interest classifier using the Hierarchical Discriminant Component Analysis (HDCA) algorithm. This algorithm linearly combines EEG electrodes in such a way that maximizes the difference between the two conditions of positive and negative [47, 46, 52].

Choosing an optimal target prevalence is an important parameter to ensure subjects maintain focus during RSVP. When the prevalence is too high or too low, the level of engagement drops, thus increasing the chances of missing a target detection. In the work of Gerson et al. [25] the target prevalence was 2% for five hertz presentation speed, however experiments during development of the C3Vision system showed that a 4% prevalence of targets improves the time needed to calibrate the EEG interest classifier as well as its performance. We use this 4% prevalence for calibration in this research, meaning in a block of 100 images only four of them are targets.

Once the EEG interest classifier is calibrated, the RSVP setup can be used to

show new images to a subject and produce *interest scores* for each image presented. The computed EEG *interest scores* for each image, which is monotonically related to the probability of the MLO given the EEG, can be used to rank a set of presented images. If the EEG interest classifier has been calibrated well, images of interest will be ranked highly, providing a significant improvement in terms of time spent and detection performance when compared to traditional methods of image search [52, 51].

8.3.1 RSVP Setup

The RSVP setup that we utilized for this research is part of the C3Vision system. Prior to calibrating the EEG interest classifier, subjects are shown a video explaining the task and providing some information about the sonar system and the goals of the project. After the video we familiarize subjects with example target and non-target *image chips*, which are specifically not part of the calibration or testing datasets. We then conduct a few practice blocks allowing the subject see the RSVP first hand as well as allowing us a chance to correct any mistakes regarding target identification. In this practice phase, no EEG data is collected.

Once a subject is confident they can differentiate targets and non-targets we begin the calibration phase to generate the EEG interest classifier. The calibration process consists of 25 blocks where each block has 100 total *image chips* containing four targets randomly mixed in with 96 non-targets to achieve the 4% target prevalence. The 100 *image chips* per block are randomly selected from the calibration pool of 240 *image chips*, which are 100 by 50 pixel images pulled from the same 975 image training set described in Section 8.2 for training the Haar-like feature classifier. There is no overlap in data between this calibration phase and the testing data presented in this research.

The *image chips* are presented to the subject at a speed of five frames per second, or five hertz. After each round the subject is presented with a graphical interface providing

feedback as to the ranking assigned to the target *image chips* compared to their order of presentation. No subject scored below 70% on this calibration step before proceeding to the experiment.

During the actual experiment the *image chips* are again shown at five hertz. The experiments are split in to sections with five blocks of 100 *image chips* each. After each block, the subject has the power to control the start of the next block, allowing them to adjust for their own fatigue. When a subject finishes a section of 500 *image chips* they are all displayed in a graphical interface in order of their rankings by *interest score*. This allows the subject and the researcher to gauge the quality of the experiment as it progresses.

The EEG data collection hardware used in this experiment is the B-Alert X10 wireless headset from Advanced Brain Monitoring. This device has nine electrodes distributed over the scalp in the standardized positions of the ten-twenty electrode system [29]. Figure 8.6 shows the wireless headset in the left image and the electrode layout in the right image. Each electrode placement is labeled with a letter to represent a lobe and a number to represent a location within a hemisphere. The four lobes utilized by this sensor configuration include frontal (F), central (C), parietal (P), and occipital (O), where the PO refers to a point between the parietal and occipital lobes. The odd numbers represent locations on the left hemisphere, while even numbers represent locations on the right hemisphere. Two reference electrodes, one behind each ear on the mastoid, are used to filter out unwanted artifacts due to muscle movement.

All RSVP experiments were conducted in the same office in the Department of Computer Science Building on the University of California San Diego (UCSD) campus. The subjects each sat in the same chair situated such that their heads would be approximately two feet from the computer screen when sitting up against the edge of the desk.

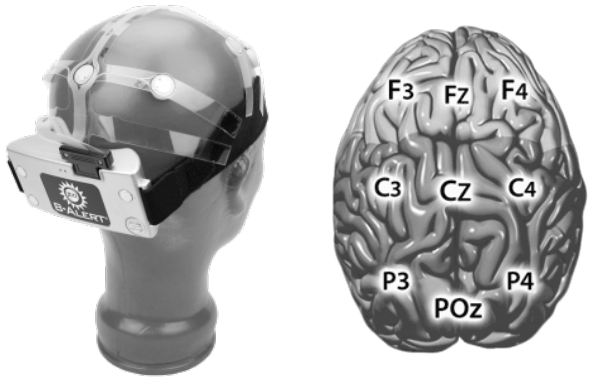


Figure 8.6. Images provided by the EEG headset manufacturer, Advanced Brain Monitoring. The left image shows the B-Alert X10 wireless headset. The right image shows the layout of the sensor nodes over the brain, where the letter indicates the lobe location and the number indicates the location within a hemisphere. The lobes used by this EEG headset include the frontal (*F*), central (*C*), parietal (*P*), and occipital (*O*) where *PO* is between the parietal and the occipital. The odd numbers represent locations on the left hemisphere and even numbers represent locations on the right hemisphere.

There were a total of 19 subjects who volunteered for the RSVP experiments from the UCSD and SSC-PAC communities. All participants had normal or corrected-to-normal vision and no history of neurological problems. There was no compensation for volunteering to be part of this experiment. Informed consent was obtained from all participants in accordance with the guidelines and approval of the UCSD Institutional Review Board.

8.3.2 Results

As a baseline for the RSVP experiments we create a preprocessed dataset of *image chips* from the 450 sonar images in our dataset. The preprocessing is very simple, splitting the sonar image in to sixteen 280 by 265 pixel regions with a 32 pixel horizontal overlap and a 20 pixel vertical overlap between regions. The result is 7,200 *image chips* to fully represent the 450 image dataset.

The RSVP experiment uses subjects with attention spans, and therefore 7,200

image chips is a large set of images for viewing. For this experiment, we only process a 245 subset of the images, or 3,920 *image chips*. Because of the overlap in windows during the preprocessing, there are 168 positive *image chips* in this RSVP data subset, referred to in the paper as *FULL* because it fully includes the images without any true preprocessing. Figure 8.7 shows two example windows from the *FULL* dataset, both including a target MLO in very different locations.

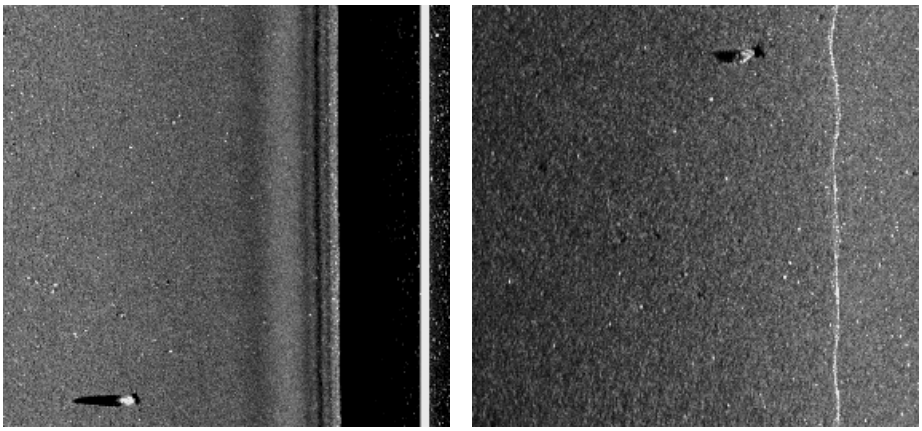


Figure 8.7. Examples of the *FULL* dataset *image chips*, which are approximately one-sixteenth of the original sonar image with 32 pixels of horizontal overlap and 20 pixels of vertical overlap. Both images show a mine of a different type in a very different location.

Figure 8.8 shows the ROC curves for the six subjects processing this *FULL* dataset for this RSVP experiment. The vertical axis shows the true positive rate (TPR), which is the number of positive *image chips* found out of the 168 in the dataset. The horizontal axis shows the number false positives per square kilometer (FP per km^2). This is the number of *image chips* on average that were classified as positive but were actually negative within a square kilometer region.

The first take-away from this figure is that, overall, the six subjects perform poorly on the *FULL* dataset. This can be attributed to the size of the *image chips* compared to the average size of the MLOs, which requires the subject to search the window in the short fifth of a second available. Similarly, the location of an MLO could be anywhere in

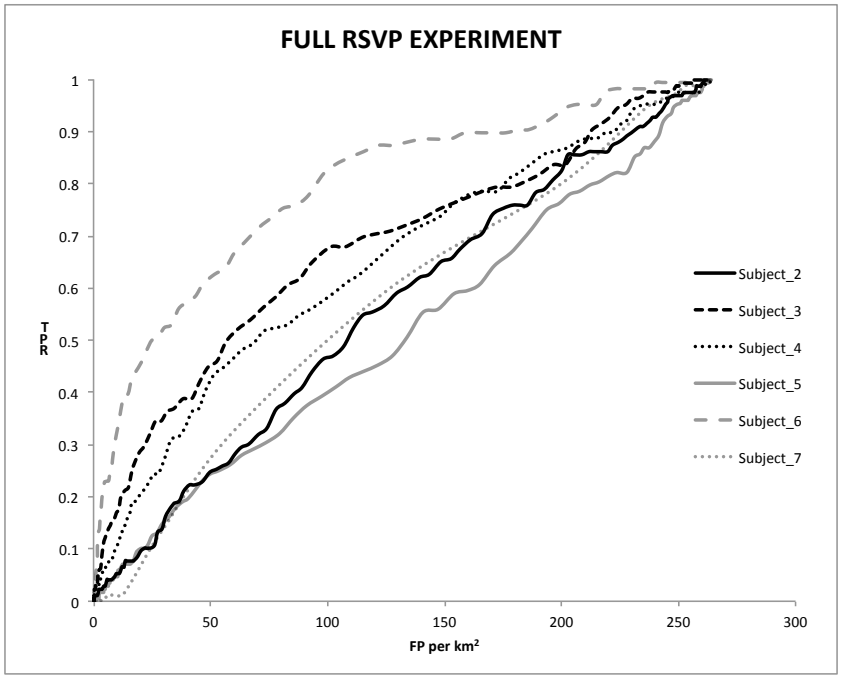


Figure 8.8. The vertical axis is true positive rate (TPR) out of the 168 *image chips* and the horizontal axis is false positives per square kilometer (FP per km^2). Shows the receiver operating characteristic (ROC) curves of the subjects processing the *FULL* dataset.

the *image chip* because of the simple preprocessing. The overlapping setup of the *FULL* dataset is likely to create a *image chip* including only a partial view of the MLO.

Another aspect of this experiment to note is the range of performance among subjects. Five of the subjects perform similarly with *Subject_6* performing substantially better. This shows that the capability of a subject to identify MLOs quickly and work well with the EEG setup is important in the results.

Overall this section shows that the RSVP experiment with the *FULL* dataset does not perform very well, especially compared to the Haar-like feature classifier performance discussed in Subsection 8.2.3. This leads to the main contribution of this chapter, which is using the Haar-like feature classifier as the preprocessing stage to prepare *image chips* for the RSVP. We present this capability and the experimental results in the next section.

8.4 Brain-Computer Interface

As we have seen in the previous section, the RSVP capability does not perform well on the *FULL* data subset and conversely the Haar-like feature classifier performs quite well on the 450 images of our complete dataset. We propose a brain-computer interface (BCI) that cascades the Haar-like feature classifier before the RSVP, similar to the C3Vision system. This means that the Haar-like feature classifier first processes the 450 sonar images and all positive regions become *image chips* that are then presented to the subjects with the C3Vision RSVP setup. This is visualized in the next section by Figure 8.16, which shows this two stage BCI system as part of the three stage system we introduce in Section 8.5.

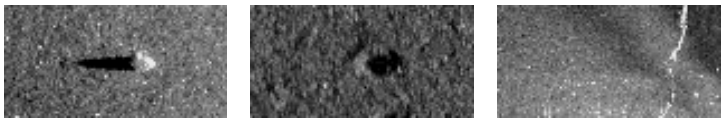


Figure 8.9. Examples of the BCI *image chips*, including a positive, negative, and known-negative example. The known-negative is a filler used to create sparsity for the versions of the experiment that do not produce a large number of negative regions.

The Haar-like feature classifier outputs a 79 by 29 pixel window for positive regions, which we increase to 100 by 50 pixels for the *image chip*. This padding around the potential positive region is meant to provide some background pixels as context with the target. Figure 8.9 shows examples of positive and negative *image chips* with the background context padding. This figure includes a positive on the left, a negative in the middle and a known-negative on the right. In this research, the known-negatives are filler *image chips* in order to create a dataset with the goal 4% prevalence of positive *image chips*.

If the padding cannot be achieved based on the location in the image then the particular region is skipped from output. Since it is not output for further processing with

the RSVP, it is marked as either a true positive (TP) or a false positive (FP) by comparison to ground truth. Similarly, it is possible that a MLO is missed by the Haar-like feature classifier, which is called a false negative (FN), and therefore is not considered by the RSVP. The total TP, FP, and FN values of the *image chips* not passed to the RSVP are still counted as part of the BCI classifiers overall performance.

The following subsections present three separate versions of this experiment, each using the Haar-like feature classifier with a different threshold for minimum neighbors. These correspond to the three points shown in Figures 8.5, where *CV-0* uses a threshold of zero, *CV-2* uses a threshold of two, and *CV-4* uses a threshold of four. As the threshold increases the number of skipped TPs remains the same while the FNs increase, leading to a reduction in positive *image chips* produced. The number of negative *image chips* produced decreases greatly with the increase of the threshold.

Table 8.1. Dataset metrics for the three BCI experiments. The skipped columns show the true positive (TP), false positive (FP), and false negative (FN) regions that are skipped after the first stage Haar-like feature classifier. The output columns show the number of *image chips* of each type.

	Skipped			Output *image chips*			
	TP	FP	FN	Positives	Negatives	Known-Negatives	Total
CV-0	1	64	4	146	4,238	0	4,384
CV-2	1	10	7	143	727	2,580	3,450
CV-4	1	6	16	134	281	2,985	3,400

Table 8.1 shows the metrics for the intermediate datasets output from each classifier version. As described, the TP, FP, and FN values that are skipped become part of the BCI's overall performance values. For *CV-2* and *CV-4* the number of positive and negative regions produced is so low that a large number of negative regions, which are referred to as known-negatives, are added to the RSVP data subset in order to keep

positive *image chips* at approximately 4% of the total *image chips*. This target prevalence is essential for the performance of the RSVP algorithm, as explained in Section 8.3.

8.4.1 CV-0 Experiment

The *CV-0* experiment uses the Haar-like feature classifier with threshold of zero to process the 450 sonar images. This classifier version creates the largest dataset for the RSVP since it is the most liberal, allowing all positively labeled *image chips* through regardless of the classification of neighboring regions. It also has the lowest number of false negatives that are not passed to the RSVP.

When processing the images, there are naturally some MLOs that are not marked

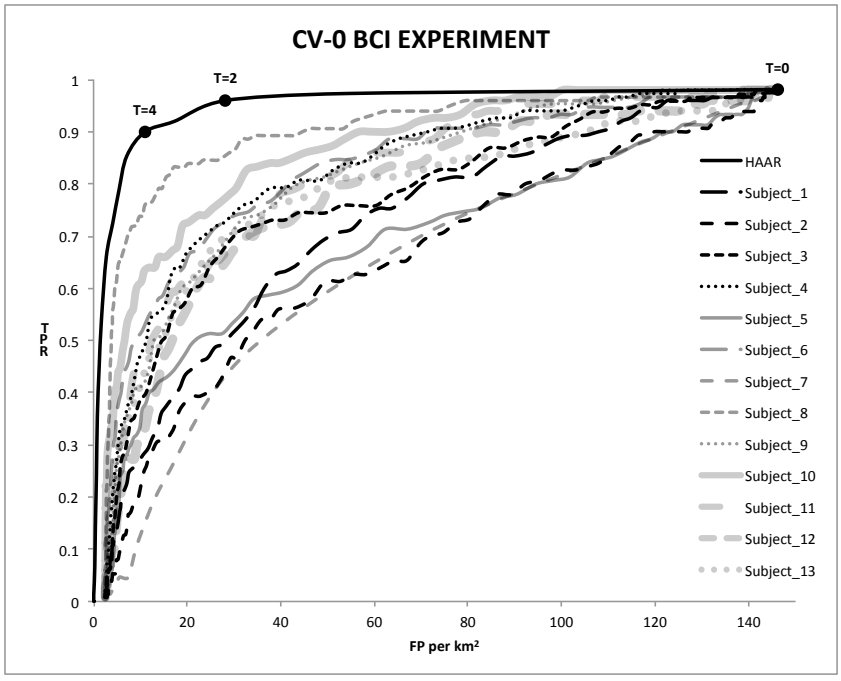


Figure 8.10. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). Shows the receiver operating characteristic (ROC) curves of the BCI experiments with the Haar-like feature classifier at the zero-threshold and various subjects using the RSVP setup. The Haar-like feature classifier ROC curve, *HAAR*, is also included for comparison. The three threshold points of interest are highlighted on the curve.

as positive by the classifier. With a threshold of zero, the classifier has three such FNs that are not considered by the RSVP experiment. Also, because the *image chips* produced are 100 by 50 pixels to include background context, one TP and 64 FP regions are skipped and not presented in the RSVP. The resulting *CV-0* dataset includes 4,384 *image chips*, including 146 positive and 4,238 negative. All of this information is summarized in the first row of Table 8.1.

The performance of the BCI classifier using the zero-threshold version of the Haar-like feature classifier and thirteen subjects with the RSVP process is shown in Figure 8.10, with a zoomed in view near the zero-threshold point shown in Figure 8.11. For each figure the vertical axis is the true positive rate (TPR), which is the number of MLO targets found by the entire BCI classifier out of the 150 targets present in the data. The horizontal axis is false positives per square kilometer (FP per km^2) based on the the known range of the images in the dataset.

The full view of the curves shown in Figure 8.10 shows an improvement in performance over the *FULL* RSVP results presented in Figure 8.8. However, it is clear that the Haar-like feature classifier alone is more accurate than any subject using the *CV-0* version of the BCI. Despite this less accurate performance, the zoomed in view of Figure 8.11 shows some potential for the BCI classifier. Notice that many subjects are able to maintain a high TPR while reducing the FP per km^2. Specifically, the curve created by *Subject_10* reduces the FP per km^2 by approximately 50.

8.4.2 CV-2 Experiment

The *CV-2* experiment uses the Haar-like feature classifier with a threshold of two to process the 450 sonar images. It is obviously more conservative than the zero-threshold version, because two other positive images must be present for a positive label. This means, in terms of skipped images, it produces less FPs, totaling ten, and more FNs, up

to seven, when compared to the *CV-0* experiment in Subsection 8.4.1. The number of TPs stays exactly the same.

There are substantial changes to the number of *image chips* output by the two-threshold classifier. The increase in conservativeness means that the classifier only outputs 870 total *image chips*, with 143 of them positive and 727 of them negative. Due to the nature of the RSVP experiment, the percentage of positives out of total windows is optimal around 4%, as explained in Section 8.3. In order to achieve this ratio, we add 2,580 known-negatives. These are *image chips* that are just filler images without a target or anything similar present. An example known-negative is shown in Figure 8.9 as the third image. The result of the additional images in the *CV-2* dataset is 3,450 *image chips*

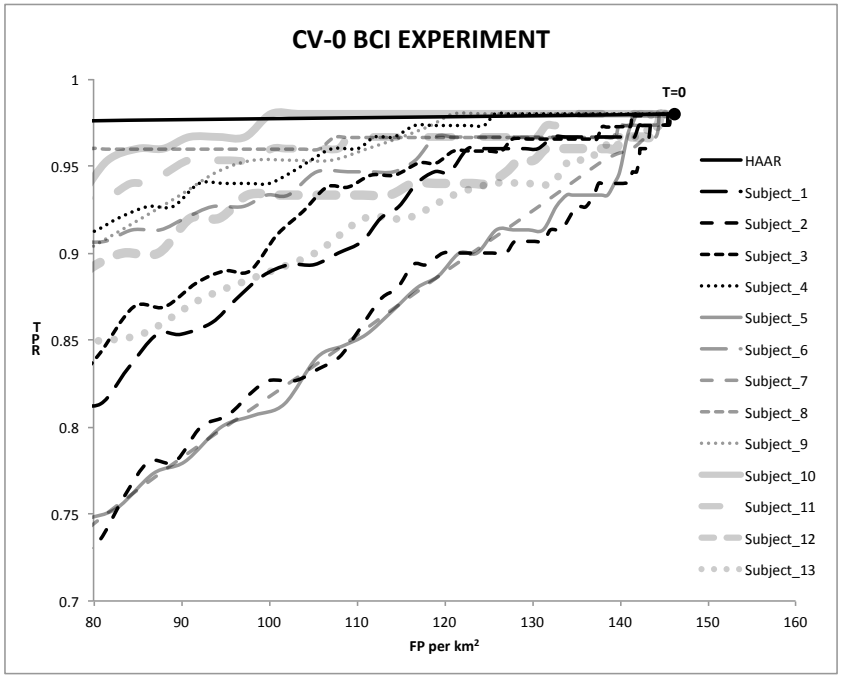


Figure 8.11. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). This is zoomed in to a smaller range in order to show the results near the labeled threshold point of zero. Shows the receiver operating characteristic (ROC) curves of the BCI experiments with the Haar-like feature classifier at zero-threshold and various subjects using the RSVP setup. The Haar-like feature classifier ROC curve, *HAAR*, is also included for comparison.

with 143 of them positive. All of this data metrics is shown in row two of Table 8.1.

Figure 8.12 shows the ROC curves for the six BCI experiments including the two-threshold version of the Haar-like feature classifier and the the RSVP process for six different subjects. There is a zoomed in view of these same curves in Figure 8.13, with a focus around the *T=2* threshold point of the Haar-like feature classifier. For each figure the vertical axis is the true positive rate (TPR), which is the number of MLO targets found by the entire BCI classifier out of the 150 targets present in the data. The horizontal axis is false positives per square kilometer (FP per km^2) based on the range of the images in the dataset.

The regular view of the curves in Figure 8.12 shows a BCI capability that more

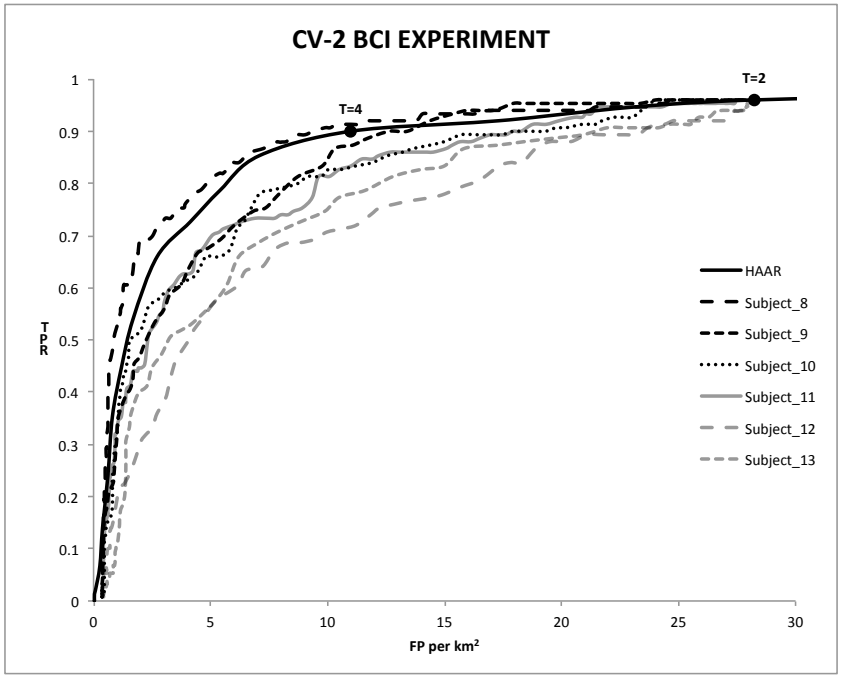


Figure 8.12. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). Shows the receiver operating characteristic (ROC) curves of the BCI experiments with the Haar-like feature classifier at the two-threshold and various subjects using the RSVP setup. The Haar-like feature classifier ROC curve, *HAAR*, is also included for comparison. Two of the threshold points of interest are highlighted on the curve.

closely rivals the performance of the Haar-like feature classifier alone, with the BCI results for *Subject_9* outperforming the *HAAR* classifier. Looking at the zoomed in view of Figure 8.13 we see that a couple other BCI classifiers are able to outperform the *HAAR* classifier at some FP per km^2 values. This shows some capability improvement and great potential with higher performing subjects or with more conservative first stage classifiers.

8.4.3 CV-4 Experiment

Similar to the previous two subsections, this subsection presents the results for the BCI experiments with the first step being the Haar-like feature classifier using a threshold of four. This is the most conservative version of the classifier that we use for

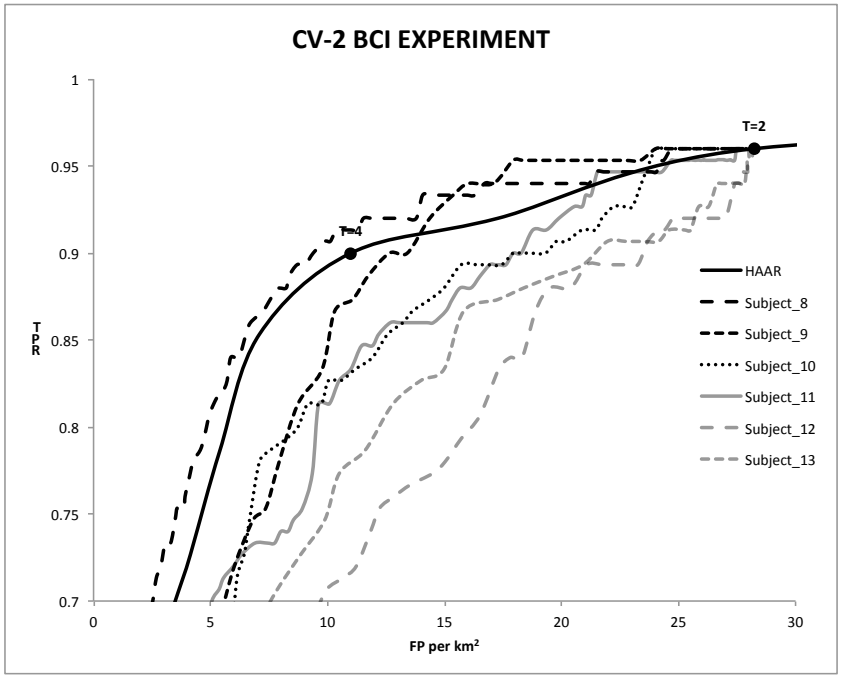


Figure 8.13. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). This is zoomed in to a smaller range in order to show the results near the labeled threshold point of two. Shows the receiver operating characteristic (ROC) curves of the BCI experiments with the Haar-like feature classifier at threshold two and various subjects using the RSVP setup. The Haar-like feature classifier ROC curve, *HAAR*, is also included for comparison.

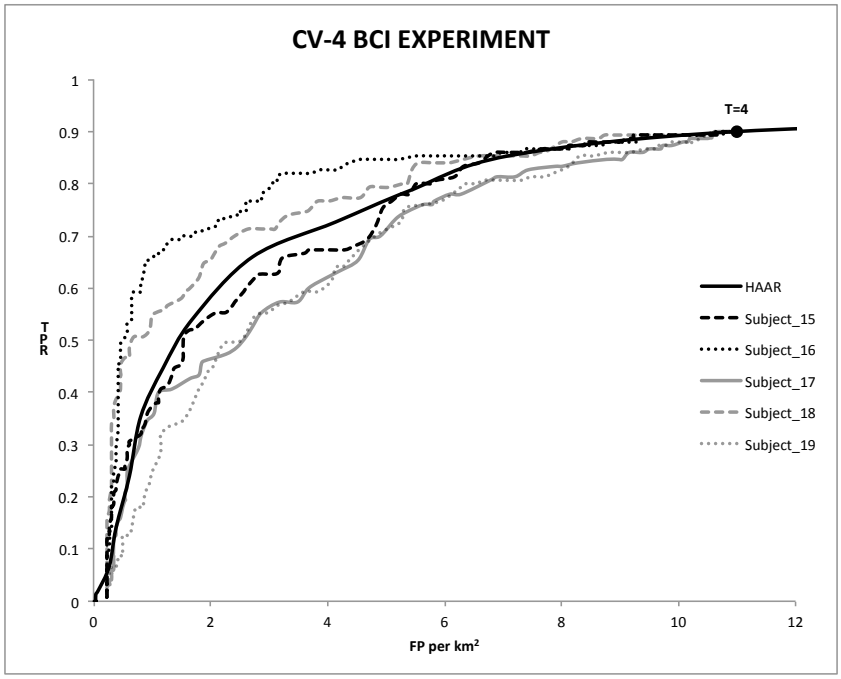


Figure 8.14. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). Shows the receiver operating characteristic (ROC) curves of the BCI experiments with the Haar-like feature classifier at threshold four and various subjects using the RSVP setup. The Haar-like feature classifier ROC curve, *HAAR*, is also included for comparison. The one threshold point of interest is highlighted on the curve.

this research. The conservativeness results in the lowest number of skipped FPs at six and the highest number of skipped FNs, totaling sixteen. The sixteen skipped FNs means that the best TPR that the BCI classifier can achieve with the RSVP step is approximately 90%.

The Haar-like feature classifier at the threshold of four produces only 415 *image chips*, with 134 positive and 281 negative. In order to achieve the 4% target prevalence, we add 2,985 known-negatives to the *CV-4* dataset, reaching 3,400 total *image chips*. These are the same known-negatives as discussed in Subsection 8.4.2 and an example is shown in Figure 8.9 as the third image. All of this data metrics about the *CV-4* BCI experiment and dataset is shown in row three of Table 8.1.

The five BCI experiments with the Haar-like feature classifier at a threshold of four cascaded in to the RSVP with five different subjects is presented in Figure 8.14. The zoomed in version of the figure, shown in Figure 8.15, focuses on the threshold four point on the *HAAR* classifier curve. For both of these figures, the vertical axis is the true positive rate (TPR), which is again the number of MLO targets found by the entire BCI classifier out of the 150 possible targets. The horizontal axis is false positives per square kilometer (FP per km^2), which is calculated based on the known range in the sonar images.

The standard view of the curves in Figure 8.14 shows a BCI capability that averages around the same performance as the Haar-like feature classifier alone over all

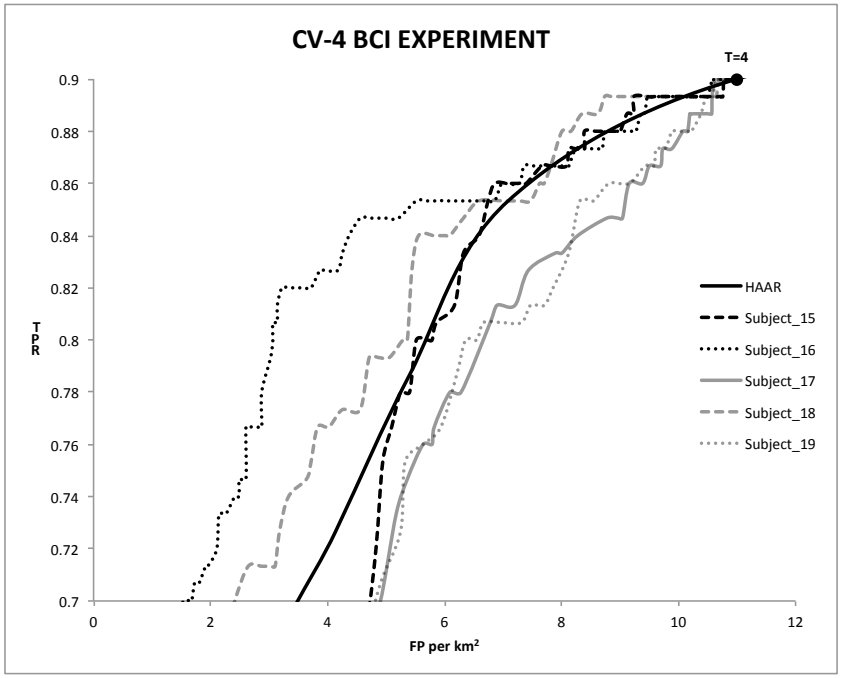


Figure 8.15. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). This is zoomed in to a smaller range in order to show the results near the labeled threshold point of four. Shows the receiver operating characteristic (ROC) curves of the BCI experiments with the Haar-like feature classifier at threshold four and various subjects using the RSVP setup. The Haar-like feature classifier ROC curve, *HAAR*, is also included for comparison.

the BCI results. The BCI results for *Subject_16* and *Subject_18* substantially outperform the *HAAR* classifier. Looking at the zoomed in view of Figure 8.15 we see that a couple other BCI classifiers are able to marginally outperform the *HAAR* classifier for a range of FP per km^2 values.

This scenario with Haar-like feature classifier at threshold four provides the best performance of the three, showing that certain subjects can be used to create a BCI classifier with substantial improvements over the computer vision technique alone, but that the system is very dependent on subject capabilities at recognizing MLOs and interacting with the RSVP system.

8.5 Brain-Computer Interface with Support Vector Machine Classifier

The previous section presents a BCI that uses the Haar-like feature classifier as a first stage, cascading the output in to the RSVP process as *image chips* and outputting a final label for locations in the full sonar image. This has mixed success with some subjects improving the capability and some subjects performing worse than the Haar-like feature classifier alone. This section proposes training a classifier that uses the Haar-like feature from the computer vision domain with the EEG *interest score* feature from the human vision domain. These two feature types are combined in a feature vector to train a support vector machine (SVM) classifier, which becomes a third stage in the BCI pipeline.

Figure 8.16 shows a diagram visualizing the difference between the system chain for the two stage BCI introduced in the previous section and the three stage BCI introduced in this section. The two stage BCI system uses the EEG scores to choose labels for the *image chips*, while the three stage BCI system passes the EEG *interest scores* and the *image chips* to the SVM classifier, which then produces the label.

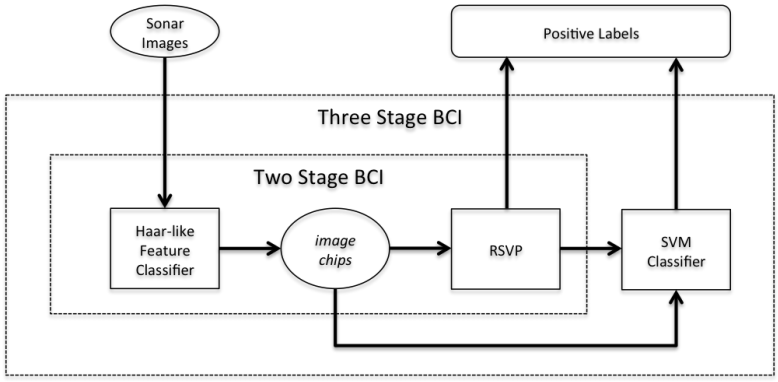


Figure 8.16. Visualization of the different BCI chains introduced in this chapter. The two stage BCI, introduced in Section 8.4, has the Haar-like classifier as stage one and the RSVP system as stage three. The three stage BCI, introduced in this section, adds the third stage of an SVM classifier.

In order to run this experiment, we divide our 450 image set in to training and testing sets of 225 images each. This way we can use the training set to create our SVM and use the testing set to compare this BCI classifier including the SVM to the Haar-like feature classifier presented in Section 8.2 and the best two stage BCI classifier presented in Section 8.4.

Before we explain our training setup and experiment results, it is important to understand support vector machines (SVMs). An SVM is a supervised learning algorithm that, given labeled examples, outputs an optimal hyperplane to divide the examples in to positive and negative [12] . The optimal hyperplane is the one that best splits the training examples with the largest distance from the nearest example point.

Figure 8.17 shows two similar visualizations of example data in multidimensional space and the optimal hyperplane selected by the SVM training. The points are shown in two dimensional space for ease of explanation. The left image shows a scenario where the data is fully separable with a hyperplane. The optimal hyperplane shown is the one that creates the largest margin, which is two times the distance from the hyperplane to the nearest examples. The right image has two additional points that create a scenario

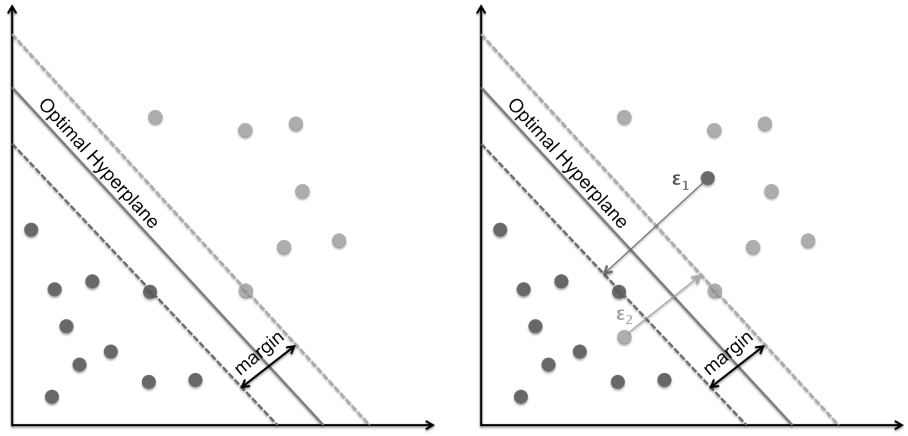


Figure 8.17. These example graphs visualize the result of the SVM training in the form of an optimal hyperplane. The left image shows a fully separable dataset, where the hyperplane function only maximizes the margin. The right image shows a dataset that cannot be separated, where the hyperplane function maximizes the margin and minimizes the total error.

where the data cannot be fully separated. In this case the algorithm must still try to maximize the margin while at the same time minimizing the total error. The errors for the misclassified data points are labeled by ε_1 for the dark point and ε_2 for the light point.

The SVM training algorithm we utilize automatically chooses the best parameters using cross-validation, where the training data is split in to ten subsets with one for training and the remaining for testing. The training repeats for each combination of one subset for training and nine for testing to select the best parameters while finding the optimal hyperplane. We use a radial basis function (RBF) as the kernel for the SVM training algorithm, which is a function that only depends on the distance from a single point such as the origin.

For this research we train SVMs on two training datasets, one composed of *image chips* created by the Haar-like feature classifier threshold set to two, called *SVM-TRAIN-2*. The other training set is composed of *image chips* created by the classifier with the threshold set to four, called *SVM-TRAIN-4*. The number of training *image chips* for the

Table 8.2. Training dataset metrics for the two SVM classifiers. Shows the number of positive and negative *image chips*, as well as the total.

	Positives	Negatives	Total
SVM-TRAIN-2	72	352	422
SVM-TRAIN-4	66	141	207

SVMs are shown in Table 8.2.

When training the SVMs on the *SVM-TRAIN-2* data we use a feature vector including both computer vision and human vision features. The computer vision features are the 34 Haar-like features from the Haar-like feature classifier presented in Section 8.2 and the human vision features are the six EEG *interest score* features corresponding to six subjects as presented in Section 8.4. Similarly when training the SVMs on the *SVM-TRAIN-4* data, the feature vector contains the same 34 Haar-like features plus the five EEG *interest score* features corresponding to the five subjects shown these *image chips*.

Once the SVMs are trained on each dataset, we can test the BCI experiment with an SVM classifier as the third stage. Table 8.3 shows the testing data for the two versions with Haar-like feature classifier thresholds of two and four, called *SVM-TEST-2* and

Table 8.3. Testing dataset metrics for the two BCI with SVM experiments. The skipped columns show the true positive (TP), false positive (FP), and false negative (FN) images that are skipped after the first stage Haar-like feature classifier. The output columns show the number of *image chips* of each type that cascade to the RSVP to produce *interest scores* for final classification by the SVM classifier.

	Skipped			Output *image chips*		
	TP	FP	FN	Positives	Negatives	Total
SVM-TEST-2	0	6	4	70	375	445
SVM-TEST-4	0	4	7	68	141	209

SVM-TEST-4 respectively. In the first section we show the skipped targets that will be included in our totals after the final classification. We see that no TPs were skipped, but a few FPs and FNs are skipped for each. The *image chips* that are cascaded in to the RSVP setup have similar distribution to the training data. The RSVP stage calculates *interest scores*, which are then used as part of the SVM classifier input to give the final label.

Figure 8.18 shows the results of the BCI with SVM experiment on the *SVM-TEST-2* dataset created when the Haar-like feature classifier has a threshold of two. The vertical axis is the true positive rate (TPR), which is the number of MLO targets found by the entire BCI classifier out of the 74 possible targets. The horizontal axis is false positives

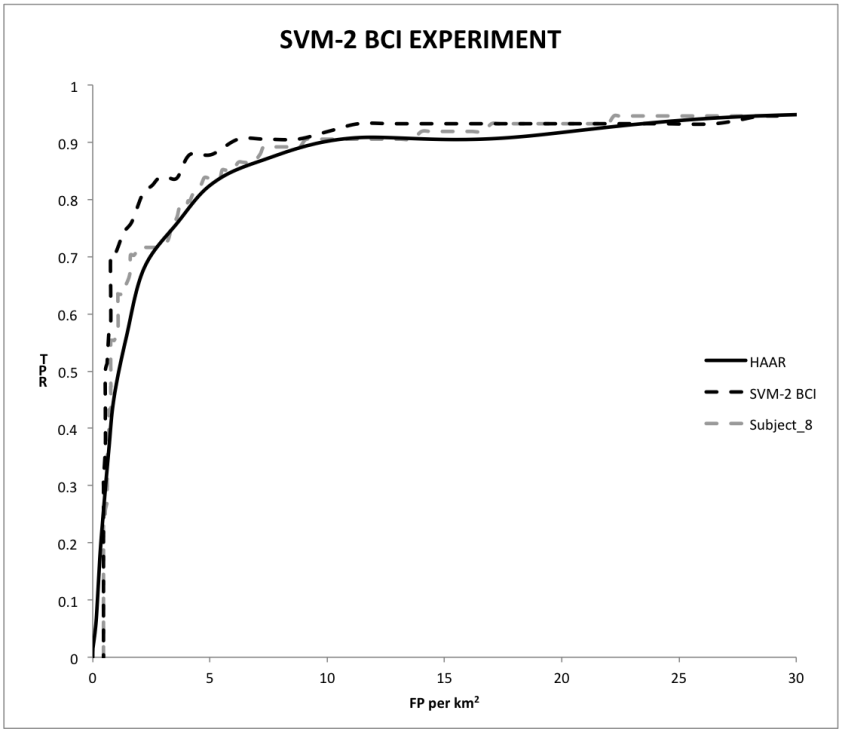


Figure 8.18. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). This compares the receiver operating characteristic (ROC) curves of the three stage BCI experiment concluding with the SVM classifier, the Haar-like feature classifier, *HAAR*, and the two stage BCI using Subject 8, *Subject_8*. For all curves, the Haar-like feature classifier portion uses a threshold of two.

per square kilometer (FP per km^2), which is calculated based on the known range in the sonar images.

The curves in this figure are all created by testing classifiers on the *SVM-TEST-2* dataset. The *HAAR* curve uses the Haar-like feature classifier from Section 8.2 and the *Subject_8* curve uses the best BCI classifier corresponding to *CV-2* from Section 8.4. Notice that the *SVM-2 BCI* curve created by the three stage BCI outperforms both previous classifiers.

Figure 8.19 shows similar results for the three stage BCI, but with Haar-like feature classifier at a threshold of four. The ROC curves are created by testing on the

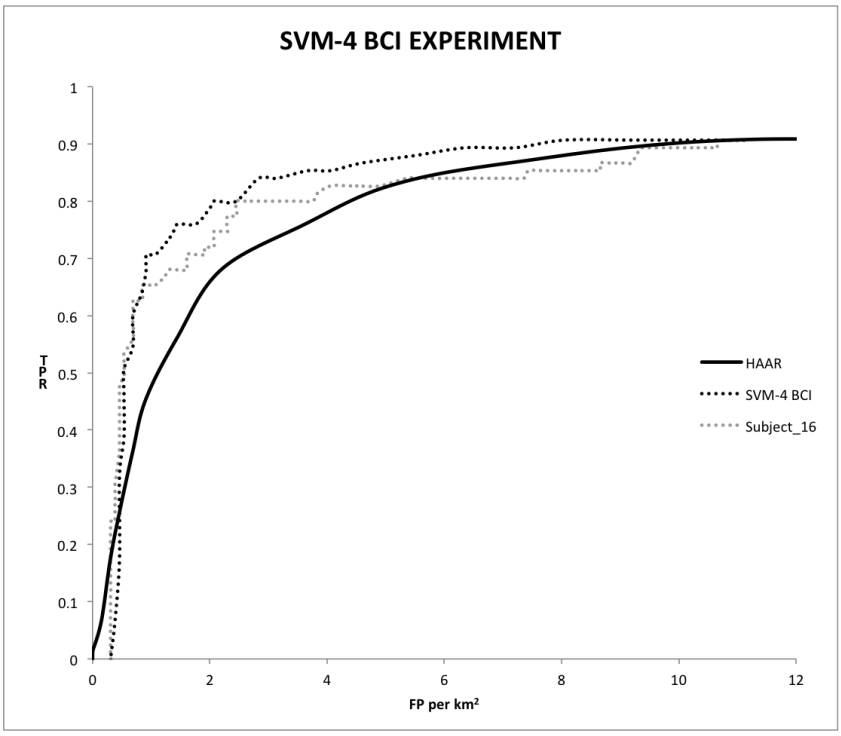


Figure 8.19. The vertical axis is true positive rate (TPR) and the horizontal axis is false positives per square kilometer (FP per km^2). This compares the receiver operating characteristic (ROC) curves of the three stage BCI experiment concluding with the SVM classifier, the Haar-like feature classifier, *HAAR*, and the two stage BCI using Subject 16, *Subject_16*. For all curves, the Haar-like feature classifier portion uses a threshold of four.

SVM-TEST-4 test set. Again the vertical axis is the true positive rate (TPR), which is the number of MLO targets found by they entire BCI classifier out of the possible 75 that exist. The horizontal axis is false positives per square kilometer (FP per km^2), as in previous figures.

Notice the same general results where the three stage BCI using the SVM classifier outperforms the other classifiers. The *HAAR* curve uses the Haar-like feature classifier presented in Section 8.2 with a threshold of four and the *Subject_16* curve uses corresponding two stage BCI with the best performing subject. Another important element to take notice of is that the amount of data used to train these SVM classifiers is about a quarter of the data used to train the *HAAR* classifier, as described in Section 8.2.

This experiment is an initial attempt at a novel concept of training a classifier using the computer vision feature and the human vision feature in the same feature vector. It shows great promise, but with limited data and breadth of experimentation, it allows space for further investigation.

8.6 Conclusion

This chapter introduces a brain-computer interface (BCI) approach to the detection of mine-like objects (MLOs) in side scan sonar imagery. The BCI system combines the complementary benefits of computer vision and human vision. We explain in depth the Haar-like feature classifier that represent the computer vision component and present its performance receiver operating characteristic (ROC) curve. We then provide detailed background on the rapid serial visual presentation (RSVP) process that uses electroencephalography (EEG) based *interest scores* to classify images and we present its performance ROC curve for six subjects.

The first BCI concept that we introduce uses the Haar-like feature classifier cascaded in to the RSVP process. We run experiments on this BCI system with three

variations of the Haar-like feature classifier and multiple subjects per experiment. The results show that the subject processing the images and the conservativeness of the Haar-like feature classifier greatly affect the performance. In the end, we see improvement over the Haar-like feature classifier alone for some subjects and consistent improvement over the RSVP classification without any preprocessing.

The second BCI concept is set up the same as the first, with an additional stage that further combines the computer vision and human vision capabilities. This third stage is a support vector machine (SVM) classifier trained on the Haar-like features and EEG *interest score* features. We show that this BCI system is able to provide performance improvements over the Haar-like feature classifier alone and the best two stage BCI subject performance.

This is the first use of BCI systems using EEG interest classifiers and RSVP on the problem of mine-like object detection in side scan sonar. The combination of computer vision and human vision is a logical collaboration and we show that there is great potential for this approach to improve performance for this task.

This chapter, in full, has been submitted for publication of the material as it may appear in IEEE Journal of Oceanic Engineering, Barngrover, Chris; Althoff, Alric; DeGuzman, Paul; Kastner, Ryan, 2014. There are small changes in format and phrasing as a chapter within this larger paper. The dissertation author was the primary investigator and author of this paper.

Chapter 9

Conclusion

This paper addresses the problem of automating the detection of mine-like objects (MLOs) in side scan sonar imagery. This is a difficult task, which has been heavily researched over the years without a definitive solution that outperforms the manual approach in real world scenarios. We present a series of approaches and experiments for this problem centered on the use of GentleBoost feature selection classifiers.

Before introducing the techniques considered in this research, we provide background on the side scan sonar imagery and a survey of the research conducted on this topic previously. This provides a foundation of information for the approaches and experiments in the chapters to follow.

We present a comparison of training on semi-synthetic versus real world data with GentleBoost feature selection classifiers, showing that for two different feature classifiers, the semi-synthetically trained classifiers can provide reasonable expectation for the performance when trained on real data.

We run experiments training and testing GentleBoost single-feature classifiers on six different feature types, finding that the Haar-like feature classifier performs the best. We then propose a GentleBoost based multi-feature selection framework, which is structured such that any dataset and any group of features can be used for training and classification. The concept allows the feature selection algorithm to select from a

pool of features containing multiple feature types. From this research comes the optimal combination of the six features into a multi-feature classifier containing Haar-like features, speeded up robust features (SURF), and shadow features. The aptly named *HAAR + SURF + SHADOW* classifier improves the true positive rate by 12.69% up to 88.56% at approximately 3.0 false positives per km^2 and under other parameters an increase of 3.79% up to 94.56% while reducing false positives per km^2 by 1.74.

We also present two limited experiments attempting to improve performance. The tiered classifiers concept trains a secondary classifier on the output of a more liberal first classifier. The result is a two tier classifier that reduces false positives for lower true positive rates. The multiple instance learning (MIL) approach trains a classifier on a bag of examples rather than on a single feature. This technique shows great potential for future efforts, achieving improved true positive rates at higher false positive rates.

Finally, we introduce two brain-computer interface (BCI) systems that combine the benefits computer vision and human vision. One BCI uses the Haar-like feature classifier as a first stage cascaded in to a human processing second stage. The other adds a third stage that employs a novel support vector machine (SVM) classifier based on the Haar-like feature and human *interest score* feature from multiple subjects.

The common thread for this research is the problem of automatically detecting mine-like objects (MLOs) in side scan sonar imagery. Overall, our GentleBoost feature selection classifier variations result in performance improvement for this problem, providing contributions to the oceanic engineering community as well as to machine learning, computer vision, and neurosciences.

Bibliography

[1] M. An, J. Tory Cobb, B. Shenefelt, and R. Tolimieri. Advances in group filter applications to sea mine detection. In *OCEANS 2006*, pages 1 –5, sept. 2006.

[2] Tom Aridgides and Manuel Fernández. Image-based atr utilizing adaptive clutter filter detection, llrt classification, and volterra fusion with application to side-looking sonar. In *Proc. of SPIE Vol*, volume 7664, pages 76640R–1, 2010.

[3] Mark W. Atherton. *Echoes and Images*. OysterInk Publications, 2011.

[4] J. Aulinas, M. Carreras, X. Llado, J. Salvi, R. Garcia, R. Prados, and Y.R. Petillot. Feature extraction for underwater visual slam. In *OCEANS, 2011 IEEE - Spain*, pages 1–7, 2011.

[5] Boris Babenko, Piotr Dollár, Zhuowen Tu, and Serge Belongie. Simultaneous learning and alignment: Multi-instance and multi-pose learning. In *Workshop on Faces in'Real-Life'Images: Detection, Alignment, and Recognition*, 2008.

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417, 2006.

[7] Nima Bigdely-Shamlo, Andrey Vankov, Rey R Ramirez, and Scott Makeig. Brain activity-based image classification from rapid serial visual presentation. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 16(5):432–441, 2008.

[8] W.D. Chesterman, P.R. Clynick, and A.H. Stride. An acoustic aid to sea bed survey. *Acustica*, 8:285–290, 1958.

[9] C.M. Ciany, W.C. Zurawski, G.J. Dobeck, and D.R. Weilert. Real-time performance of fusion algorithms for computer aided detection and classification of bottom mines in the littoral environment. In *OCEANS 2003. Proceedings*, volume 2, pages 1119 – 1125 Vol.2, sept. 2003.

[10] Daniel T. Cobra, Alan V. Oppenheim, and Jules S. Jaffe. Geometric distortions in side-scan sonar images: A procedure for their estimation and correction. *Journal of Oceanic Engineering*, 17(3):252–268, July 1992.

[11] E. Coiras, P.-Y. Mignotte, Y. Petillot, J. Bell, and K. Lebart. Supervised target detection and classification by training on augmented reality data. *Radar, Sonar Navigation, IET*, 1(1):83 –90, feb. 2007.

[12] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[13] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[14] Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1):31–71, 1997.

[15] Gerald J Dobeck. Fusing sonar images for mine detection and classification. In *AeroSense'99*, pages 602–614. International Society for Optics and Photonics, 1999.

[16] Gerald J Dobeck. *The k-nearest neighbor attractor-based neural network and the optimal linear discriminatory filter classifier*. IEEE, 2006.

[17] Gerald J Dobeck, John C Hyland, and Le'derick Smedley. Automated detection and classification of sea mines in sonar imagery. In *AeroSense'97*, pages 90–110. International Society for Optics and Photonics, 1997.

[18] D. T. Donovan, A. H. Stride, and A. J. Lloyd. An acoustic survey of the sea floor south of dorset and its geological interpretation. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 244(712):pp. 299–330, 1961.

[19] E. Dura, J. Bell, and D. Lane. Superellipse fitting for the recovery and classification of mine-like shapes in sidescan sonar images. *Oceanic Engineering, IEEE Journal of*, 33(4):434 –444, oct. 2008.

[20] Rogerio Feris, James Petterson, Behjat Siddiquie, Lisa Brown, and Sharath Pankanti. Large-scale vehicle detection in challenging urban surveillance environments. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 527–533. IEEE, 2011.

[21] B.W. Flemming. Side-scan sonar: A practical guide. *International Hydrographic Review*, 1976.

[22] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[23] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *The annals of statistics*, 28(2):337–407, 2000.

[24] M. Gendron, M. Lohrenz, and J. Dubberley. Automated change detection using synthetic aperture sonar imagery. In *OCEANS 2009, MTS/IEEE Biloxi - Marine Technology for Our Future: Global and Local Challenges*, pages 1 –4, oct. 2009.

[25] Adam D Gerson, Lucas C Parra, and Paul Sajda. Cortically coupled computer vision for rapid image search. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 14(2):174–179, 2006.

[26] W.D. Hackmann. *Seek & Strike: Sonar, Anti-submarine warfare, and the Royal Navy, 1914-1954*. HMSO, London, 1984.

[27] Paul Hollesen, Warren A. Connors, and Thomas Trappenberg. Comparison of learned versus engineered features for classification of mine like objects from raw sonar images. *Advances in Artificial Intelligence*, pages 174–185, 2011.

[28] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187, 1962.

[29] Herbert Henri JASPER. The ten twenty electrode system of the international federation. *Electroencephalography and clinical neurophysiology*, 10:371–375, 1958.

[30] Michael J Jones and Daniel Snow. Pedestrian detection using boosted features over many frames. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE, 2008.

[31] Ashish Kapoor, Pradeep Shenoy, and Desney Tan. Combining brain computer interfaces with vision for object categorization. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[32] N. Klausner, M.R. Azimi-Sadjadi, and J.D. Tucker. Underwater target detection from multi-platform sonar imagery using multi-channel coherence analysis. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 2728 –2733, oct. 2009.

[33] F. Langner, C. Knauer, W. Jans, and A. Ebert. Side scan sonar image resolution and automatic object detection, classification and identification. In *OCEANS 2009. Proceedings*, May 2009.

[34] Donghui Li, M.R. Azimi-Sadjadi, and M. Robinson. Comparison of different classification algorithms for underwater target discrimination. *Neural Networks, IEEE Transactions on*, 15(1):189 –194, jan. 2004.

[35] Shengcai Liao, Xiangxin Zhu, Zhen Lei, Lun Zhang, and Stan Z Li. Learning multi-scale block local binary patterns for face recognition. In *Advances in Biometrics*, pages 828–837. Springer, 2007.

[36] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–900. IEEE, 2002.

[37] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[38] Max Mignotte, Christophe Collet, Patrick Perez, and Patrick Bouthemy. Sonar image segmentation using an unsupervised hierarchical mrf model. *IEEE Trans. Image Processing*, 9:1216–1231, 1998.

[39] Stefan Munder and Dariu M Gavrila. An experimental study on pedestrian classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1863–1868, 2006.

[40] N. Intrator N. Neretti and Q. Q. Huynh. Target detection in side-scan sonar images: expert fusion reduces false alarms. *Submitted.*, 2002.

[41] M. Neumann, C. Knauer, B. Nolte, W. Jans, and A. Ebert. Target detection of man made objects in side scan sonar images segmentation based false alarm reduction. In *Acoustics 08. Proceedings*, 2008.

[42] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59, 1996.

[43] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[44] Aude Oliva. Gist of the scene. *Neurobiology of attention*, 696:64, 2005.

[45] C.P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Computer Vision, 1998. Sixth International Conference on*, pages 555 –562, jan 1998.

[46] Lucas C Parra, Christoforos Christoforou, Adam D Gerson, Mads Dyrholm, An Luo, Mark Wagner, Marios G Philiastides, and Paul Sajda. Spatiotemporal linear decoding of brain state. *Signal Processing Magazine, IEEE*, 25(1):107–115, 2008.

[47] Lucas C Parra, Clay D Spence, Adam D Gerson, and Paul Sajda. Recipes for the linear analysis of eeg. *Neuroimage*, 28(2):326–341, 2005.

[48] C. Rao, K. Mukherjee, S. Gupta, A. Ray, and S. Phoha. Underwater mine detection using symbolic pattern analysis of sidescan sonar images. In *American Control Conference, 2009. ACC '09.*, pages 5416 –5421, june 2009.

[49] Scott Reed, Yvan Petillot, and Judith Bell. An automatic approach to the detec- tion and extraction of mine features in sidescan sonar. *IEEE J. Ocean. Eng.*, 28(1):90 – 105, Jan. 2003.

[50] P. Saisan and S. Kadambe. Shape normalized subspace analysis for underwater mine detection. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 1892 –1895, oct. 2008.

[51] Paul Sajda, Eric Pohlmeyer, Jun Wang, Barbara Hanna, Lucas C Parra, and Shih-Fu Chang. Cortically-coupled computer vision. *Brain-Computer Interfaces*, pages 133–148, 2010.

[52] Paul Sajda, Eric Pohlmeyer, Jun Wang, Lucas C Parra, Christoforos Christoforou, Jacek Dmochowski, Barbara Hanna, Claus Bahlmann, Maneesh Kumar Singh, and Shih-Fu Chang. In a blink of an eye and a switch of a transistor: cortically coupled computer vision. *Proceedings of the IEEE*, 98(3):462–478, 2010.

[53] Jamal Sawas, Yvan Petillot, and Yan Pailhas. Cascade of boosted classifiers for rapid detection of underwater objects. In *European Conference on Underwater Acoustics, ECUA*, volume 10, 2010.

[54] Jamil Sawas and Yvan Petillot. Cascade of boosted classifiers for automatic target recognition in synthetic aperture sonar imagery. In *Proceedings of Meetings on Acoustics*, volume 17, page 070074, 2013.

[55] Robert E Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine learning*, 37(3):297–336, 1999.

[56] Dana E. Skinner. *Supervised and Unsupervised learning methods for detection and classification of mine features in side scan sonar imagery*. PhD thesis, Florida State University, 2006.

[57] M.L. Somers and A.R. Stubbs. Sidescan sonar. *Communications, Radar and Signal Processing, IEE Proceedings F*, 131(3):243 –256, june 1984.

[58] Satoshi Suzuki. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[59] D.G. Tucker. Sonar and underwater acoustical engineering. *Electronics and Power*, 11(7):220 –223, july 1965.

[60] J.D. Tucker, M.R. Azimi-Sadjadi, and G.J. Dobeck. Canonical coordinates for detection and classification of underwater objects from sonar imagery. In *OCEANS 2007 - Europe*, pages 1 – 6, june 2007.

[61] M.J. Tucker and A.R. Stubbs. Narrow beam echo-ranger for fishery and geological investigations. *British Journal of Applied Physics*, 12(3), March 1961.

[62] Paul Viola and Michael Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001.

[63] Jian-Gang Wang, Jun Li, Wei-Yun Yau, and Eric Sung. Boosting dense sift descriptors and shape contexts of face images for gender recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 96–102. IEEE, 2010.

[64] Xingmei Wang, Huanran Wang, Xiufen Ye, Lin Zhao, and Kejun Wang. A novel segmentation algorithm for side-scan sonar imagery with multi-object. In *International Conference on Robotics and Biomimetics*, pages 2110 – 2114, Dec. 2007.

[65] Shuang Wei, H. Leung, and V. Myers. An automated change detection approach for mine recognition using sidescan sonar data. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 553 –558, oct. 2009.

[66] A.B. Wood, F.D. Smith, and J.A. McGeachy. A magnetostriction echo depth-recorder. *Wireless Section, Institution of Electrical Engineers - Proceedings of the*, 10(29):80 –93, june 1935.

[67] D. Yao and M. Azimi. A study of effects of sonar bandwidth for underwater target classification. *IEEE Journal of Oceanic Engineering*, 27, July 2002.

[68] Cha Zhang, John C Platt, and Paul A Viola. Multiple instance boosting for object detection. In *Advances in neural information processing systems*, pages 1417–1424, 2005.

[69] Wei Zhang, Bing Yu, Gregory J Zelinsky, and Dimitris Samaras. Object class recognition using multiple layer boosting with heterogeneous features. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 323–330. IEEE, 2005.

[70] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498. IEEE, 2006.