Maritime Threat Detection Using Probabilistic Graphical Models

Bryan Auslander[1], Kalyan Moy Gupta[1], & David W. Aha[2]

[1]Knexus Research Corporation; Springfield, VA 22153
[2]Navy Center for Applied Research in Artificial Intelligence;
Naval Research Laboratory (Code 5514); Washington, DC 20375
firstname.lastname@knexusresearch.com | david.aha@nrl navy mil

Abstract

Maritime threat detection is a challenging problem because maritime environments can involve a complex combination of concurrent vessel activities, and only a small fraction of these may be irregular, suspicious, or threatening. Previous work on this task has been limited to analyses of single vessels using simple rule-based models that alert watchstanders when a proximity threshold is breached. We claim that Probabilistic Graphical Models (PGMs) can be used to more effectively model complex maritime situations. In this paper, we study the performance of PGMs for detecting (small boat) maritime attacks. We describe three types of PGMs that vary in their representational expressiveness and evaluate them on a threat recognition task using track data obtained from force protection naval exercises involving unmanned sea surface vehicles. We found that the best-performing PGMs can outperform the deployed rule-based approach on these tasks, though some PGMs require substantial engineering and are computationally expensive.

1. Introduction

Early prediction of an evolving threatening situation is critical for maritime force protection. Methods for analyzing these situations are typically performed from one of two perspectives: (1) Wide area surveillance for post-hoc analysis, where vessels are tracked across large geographical areas (e.g., tracking international shipping vessels using the Automated Identification System (AIS)) (Bostwick et al. 2009) or (2) local area surveillance over comparatively small distances (e.g., 1000-5000 yards) for real-time maritime behavior analysis and threat detection, which is our focus in this paper. We previously studied vessel classification with video data (Gupta et al. 2009) and anomaly detection using video data extended with *synthetic* anomaly data (Auslander et al. 2011). In this paper we focus on threat detection with *real* maritime data.

In particular, here we compare algorithms for identifying threats in scenarios where a combination of unmanned sea surface vehicles (USSVs) and ground-based sensors are used to monitor maritime locations such as ports, harbors, and rivers. For example, maritime assets such as oil platforms are vulnerable to attacks from a variety of near-shore threats such as small boats. Maritime threats are assessed by watchstanders who rely on automated video surveillance systems to reduce information overload. These systems help watchstanders to monitor many concurrent contacts and provide some support for behavior analysis and threat prediction. However, state-of the-art systems for local area surveillance, which perform perimeter-based threat detection (e.g., (Lipton et al. 2002; RemoteReality 2011), are limited because they consider only the relative location of a potential threat and ignore many other features and relations among maritime vessels. We argue that maritime threats involve complex combinations of vessel types and their activities, and more sophisticated algorithms are needed to support watchstanders.

Probabilistic Graphical Models (PGMs) can be used to represent relations compactly and permit efficient inference in the presence of uncertainty (Koller and Friedman 2009). A PGM uses a declarative state representation, a probabilistic algorithm for inference, and can combine expert knowledge and accumulated data to estimate state and state transition distributions. Because PGMs can model probabilistic relations, we believe they are better suited than perimeter-based algorithms for predicting maritime threats. However, they have not previously been applied to local maritime surveillance.

In this paper, we apply and evaluate PGMs for maritime threat detection. We claim that representing and exploiting relational information enables better recognition of small vessel attacks. We evaluate three types of PGMs that vary in their representational abilities, and compare their performance against baseline (non-PGM) approaches. Our results support our claim that PGMs outperform these baseline approaches for maritime threat recognition.

We review the maritime threat detection task in §2, and introduce the PGMs we study in §3. Section 4 describes our empirical study and the results. We discuss issues pertaining to this task in §5 and conclude in §6.

2. Local Maritime Threat Detection

The detection of small vessel threats and prevention of their attacks is a crucial capability for protecting maritime personnel and assets, and its need is exemplified by the USS Cole bombing and related incidents. Navy ships and

merchant vessels can be in close proximity with smaller vessels in busy maritime locations. Unlike large vessels, small vessels do not carry AIS equipment. Therefore, wide area surveillance approaches are inappropriate for this task.

The state-of-the-art approach for *local* surveillance is a perimeter defense mechanism, which defines a perimeter (or electronic fence) about a shoreline or vessel. Given a set of rules, they trigger alerts to watchstanders when they detect other vessels that penetrate this perimeter (Lipton et al. 2002; RemoteReality 2011). This approach can substantially reduce the manual effort needed for effective video surveillance. However, it is limited; it cannot detect threats based on analysis of vessel behavior outside of this perimeter, nor reason about intent or coordinated threats.

We focus on the first of these limitations – on algorithms for detecting threats other than via only perimeter breach. It is not clear what rules exist for distinguishing threats from non-threats outside pre-defined perimeters, thus complicating an extension of the rule-based approach. Also, probabilistic algorithms may be better-suited for modeling this task. Finally, the relations (e.g., spatial, temporal, semantic) among the nearby small vessels, the maritime asset in their vicinity, and the platform used to observe these vessels may be useful for assessing whether a threat exists. Therefore, we are exploring the utility of probabilistic graphical models (PGMs) for this task. PGMs have performed well on non-maritime behavior recognition tasks (e.g., (Tran and Davis 2008; Manfredotti 2009; Lavee et al. 2009)), but to our knowledge they have not yet been applied to the task of maritime threat detection.

In recent years, autonomous USSVs have been proposed to serve a key role in force protection tasks (USV 2007). For example, they could be used to guard high-value assets in escort missions in areas of high vessel traffic. Given access to the USSV's sensors, an automated decision aid could potentially identify and track nearby vessels, detect potential threats, and (in coordination) block threatening vessels from reaching these assets. Our research focuses on the design and evaluation of maritime threat detection algorithms for these decision aids on autonomous USSVs.

3. Probabilistic Graphical Models

PGMs offer a number of benefits for modeling relations in a complex domain. They provide a compact encoding of a distribution in a multi-dimensional space, model variable independencies, and have well understood mathematical foundations. When selecting which PGMs to use in a given domain, tradeoffs must be made between feature expressiveness, learning, and inference costs.

Threat recognition in a maritime domain is a relatively unexplored application, and it is not clear which algorithms will perform well on this task. Therefore, in comparison with baseline algorithms (see §4), we examined the performance of a small but varied collection of PGMs on this task, including three that differ in their representational expressiveness: (1) Hidden Markov Models (HMMs), (2) Conditional Random Fields (CRFs), and (3) Markov Logic Networks (MLNs). We describe these in the following subsections.

3.1 Hidden Markov Models (HMMs)

An HMM is a generative model of a probabilistic sequence. An HMM model is a graph whose nodes denote hidden states and whose links denote transition probabilities from one state to another (Rabiner 1989). HMMs model the joint distribution $p(y_t,x_t)$, for observation x_t and state y_t at time t using two assumptions. First, it makes the Markov assumption: state transitions depend only on the preceding state, and are independent of all other states. Second, it assumes that each observation depends only on the current state (Sutton and McCallum 2006). The joint distribution is modeled as follows:

$$\boldsymbol{p(y,x)} = \prod_{t=1}^{T} \boldsymbol{p(y_t|y_{t-1})p(x_t|y_t)}$$

where $p(y_t|y_{t-1})$ models the transition distribution and $p(x_t|y_t)$ the observation distribution. HMM learning and inferencing is performed using the forward-backward and the Viterbi algorithms, respectively**.**

HMMs have been used successfully in many tasks such as natural language processing, speech recognition, and modeling of dynamic agents. Although they model temporal relations, they cannot compactly represent local features and spatial relations. When given multiple dependent features, an HMM becomes intractable (Sutton and McCallum 2006). Thus, HMM extensions include coupled HMMs, which represent limited relational features (Brand et al. 1997). CRFs also eliminate this limitation of HMMs, and we describe them next.

3.2 Conditional Random Fields (CRFs)

A linear chain CRF is the *discriminative* counterpart to the generative HMM and can also be used to model a sequence or an agent's actions in a temporal domain. A CRF is a discriminative model because it models the *conditional* distribution $p(y|x)$ rather than the *joint* distribution $p(y,x)$ and can reason with interdependent features. In the maritime domain, many features violate the independence assumption, which may make CRFs a more suitable model than HMMs.

Parameter learning algorithms for CRFs typically use gradient descent algorithms such as Limited-Memory BFGS (LMBFGS) (Sutton and McCallum 2006). Exact inference is also possible for linear-chain CRFs. Inference is performed using the forward-backward or Viterbi algorithms. Sutton and McCallum define linear-chain CRFs as a distribution $p(y|x)$:

$$p(y|x) = \frac{1}{Z(x)} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t)\right\}$$

where $Z(x)$ is an instance-specific normalization function

$$Z(x) = \sum_y \exp\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t)\},$$

and where Y and X are random vectors, K is the set of features, λ_k is a parameter vector, and $f_k(y_t, y_{t-1}, x_t)$ is a set of real-valued feature functions. This model leads to an exponential build up when calculating $Z(x)$, but this can be computed efficiently in the same way as in HMMs, resulting in extremely fast inferencing.

CRFs have been applied to natural language processing, bio-sequencing, and computer vision tasks. Unlike HMMs, a CRF can model local and temporal features. However, CRFs are limited in their ability to naturally model expert domain knowledge. For example, they cannot model relational spatial features such as the distances between multiple pairs of ships in a maritime domain. Markov logic networks remove this limitation, as we describe next.

3.3 Markov Logic Networks (MLNs)

MLNs combine first-order logic (FOL) with a probabilistic interpretation to represent expert domain knowledge (Domingos and Lowd 2009). In FOL, domains are defined by a set of grounded formulas. Each formula represents a hard constraint whose violation invalidates the domain knowledge. This makes FOL difficult to apply to real-world domains, whose features are rarely consistent. MLNs relax these constraints; they can model domains where constraint violations have low probability, but are not impossible. MLNs associate weights with FOL formulae, where weights represent the strength of constraint. A higher weight indicates a larger difference in the log probability between interpretations that satisfy the constraint from those that do not. Weights can be assigned manually or can be learned from example data.

An MLN is a set of pairs (F_i, w_i) where F_i is a FOL formula and w_i is a real number. Together with a finite set of constants C it defines a Markov network $M_{L,C}$ (Domingos and Lowd 2009) where:

1. $M_{L,C}$ contains one binary node for each possible grounding of each predicate appearing in the set of possible groundings L. The value of the node is 1 if the ground predicate is true, and 0 otherwise.
2. $M_{L,C}$ contains one feature for each possible grounding of each formula F_i in L. The value of this feature is 1 if the ground formula is true, and 0 otherwise. The weight of the feature is the w_i associated with F_i

From this definition, an MLN can be viewed as a template for constructing a grounded Markov network, which may vary widely in shape and size depending on its constraints. The probability distribution specified by a grounded Markov network x is represented by the following log-linear model:

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right)$$

Table 1: Qualitative Comparison of Candidate PGMs

Characteristic	HMMs	CRFs	MLNs
Representation	Feature tokens	Feature vectors	FOL with weights associated to rules
Learning Method	Generative	Discriminative	Both
Feature Types	Temporal	Temporal and local	Temporal, local, and spatial
Learns	State transition probabilities	Clique potentials	Weights for rules

where $\boldsymbol{n_i(x)}$ is the number of true groundings of F_i in x and Z is a normalization constant.

MLNs are a more general model than CRFs or HMMs, and this allows them to be applied to many of the same tasks. Also, an MLN can encode a greater amount of relational knowledge, which makes them useful in scene understanding, object recognition, and activity recognition. In the maritime domain, MLNs can more easily encode relational spatial features, which allow them to model vessel behaviors more accurately. Table 1 summarizes these three types of PGMs.

4. Empirical Study

4.1 Objectives

We hypothesized that PGMs can outperform rule-based perimeter defense models for maritime threat detection given small vessel tracks from fused USSV sensor data. Although intuitive, this has not been previously studied.

We also wanted to assess the relation between threat recognition performance and each PGM's ability to represent domain knowledge, and how the length of the situation history (see §4.4) affects relative performance.

4.2 Data

We obtained our evaluation corpus from the 2010 Trident Warrior exercise (Summer 2010). This proprietary data was provided to us by Spatial Integrated Systems, who recorded it during multi-day USSV tests in San Diego Bay. Two USSVs participated in missions that involved escorting and protecting a High Value Unit (HVU) as it moved through a channel into the open water. Periodically, two human-controlled boats would "attack" the HVU and the USSVs' task was to block the attackers. When an attack was identified, the USSV closest to the attacker would move to block while the other USSV would shift to protect the HVU from other attacks.

During these scenarios, the USSVs employed their sensors to create a shared fused situational picture. The fused data for the track of each observed vessel (i.e., the five in the scenario or others in the field of view) contained nine raw attributes: *speed, bearing, pitch, roll, time, latitude, longitude*, *id*, and *current state* (USSV only, with values such as "moving to escort" and "blocking").

However, we performed extensive preprocessing on this data prior to using it in our experiments, and used a

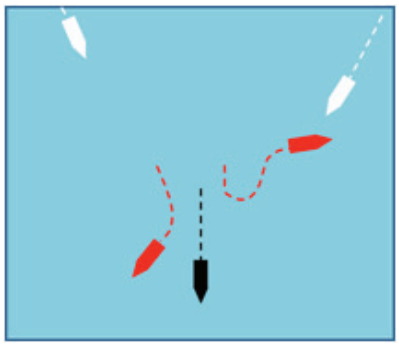

Figure 1: Annotation tool screenshot depicting attackers (White), USSVs (Red), and the HVU (black).

different set of attributes. First, we synchronized the tracks because they were recorded at different clock rates. Second, we removed noisy tracks, which were caused by sensor data error (which could yield duplicate or erroneous tracks) or non-vessels (e.g., buoys, waves). Finally, we created and applied a tool (Figure 1) to manually annotate each track instance as *Attacking*, *Cruising*, or *Escaping* depending on their perceived movement.

The raw attributes lack some useful information that the PGMs can exploit. Therefore, we computed four features per track instance (Table 2) for use by the PGMs. ***Prior Activity*** is the activity that a vessel performed in the prior time step. This value is known during training, but is predicted during testing. ***Distance to HVU*** is the tracked vessel's distance to the HVU's location. ***In Front of HVU*** denotes whether the tracked vessel is bearing on the HVU. Finally, ***Approaching HVU*** is a binary feature indicating whether the vessel's distance to the HVU is decreasing.

This produced two sets of tracks, each of which is 53 minutes in length and includes at least one attack instance, and does not break tracks that contain attacks across the two sets. The characteristics of these sets are summarized in Table 3, where a time step is 10 seconds in length.

4.3 Measures

The threat recognition task involves predicting, at each time step, whether a human-controlled boat is attacking the HVU. We used precision, recall, and F_1 (Manning et al. 2008) to assess performance. We also measured each system's run time for training and testing.

4.4 PGM tools and escort scenario models

We next describe the implementations we used for the three PGMs. Each was tested using a sliding history of observations whose size was optimized during training.

HMMs

We used MALLET (Machine Learning for LanguagE Toolkit) (McCallum 2002) to implement HMMs. We also used its sequence tagging capabilities and its implementations of the forward-backward training algorithm and the Viterbi algorithm for inference.

Table 2: Computed Features from the Fused Tracks

Features	Description	# Values
Prior Activity	Vessel's activity in the prior time step	3
Distance to HVU	Discretized distance from tracked vessel to the HVU	4
In Front of HVU	Denotes whether the tracked vessel is bearing on the USSV (200° arc)	2
Approaching HVU	Denotes whether the vessel is approaching the HVU	2

Table 3: Escort Scenario Data Sets

Details	Set 1	Set 2
Length (number of time steps)	320	320
Number of Tracks	46	53
Concurrent Tracks [min,max]	[3,24]	[3,12]
Track Instances	1148	2310
Track Instances that are Attacks	44 (3.8%)	35 (1.5%)

HMMs require a single token as input. Therefore, we concatenated the current state's feature values and provided them as input. (Only a maximum of 16 feature combinations of the possible 48 exist in our data. In future work, we will test codebook methods to fuse the features.)

During training, we created sequences of the selected window size and performed inference using them. During testing, we provided a trained HMM with a sequence of observations, from which we inferred a sequence of activities, and used the final activity as its prediction.

CRFs

We used MALLET's sequence tagger to implement our CRFs, trained them using its implementation of LMBFGS, and performed inference using the Viterbi algorithm. A CRF, unlike an HMM, can represent local features, and does not require feature concatenation.

MLNs

For MLNs, we used Alchemy (Alchemy 2011), an open source statistical relational learning and probabilistic inferencing package. Alchemy supports generative and discriminative weight learning, and multiple types of inference (e.g., MAP, belief propagation, and MCSAT). We report the results from only generative learning using MAP inference because, in our trial runs, generative learning outperformed discriminative learning.

We specified the MLNs using FOL syntax. This requires defining one or more predicates and a rule set. For example, the following rule denotes that when activity a_1 is performed at time t_1 by an agent x, then x will perform a_2 at time t_2 (where t_2 immediately succeeds t_1):

$$Activity(x,+a_1,t_1) \wedge Succ(t_2,t_1) \Rightarrow Activity(x,+a_2,t_2) \quad (1)$$

The '+' signs denote that Alchemy creates a new formula for every possible combination of the values for a_1 and a_2 that fit the type specified in their predicate declaration.

Through manual iteration, we chose MLN rules similar to Equation 1 for the features in Table 2. Alchemy grounds these rules during training. During weight learning, we input the training data and the activity labels to Alchemy. Using the sliding window, we queried Alchemy for the most likely activity for each vessel at the current time step.

Table 4: Results for Predicting Attack Instances

Algorithm	Trained on Set #1			Trained on Set #2		
	Precision	Recall	F_1	Precision	Recall	F_1
Default	0.04	1.00	0.07	0.02	1.00	0.03
Perimeter Rule	0.38	1.00	0.55	0.04	0.37	0.08
HMM	0.40	0.46	0.43	0.06	0.40	0.10
CRF	0.63	0.11	0.19	0.11	0.57	0.18
MLN	0.42	1.00	**0.59**	0.11	1.00	**0.19**

Table 5: Optimum Window Size, Training (on Set 1), and Test Times (on Set 2) in Seconds

Algorithm	Window Size	Training Time	Test Time
Perimeter Rule	N/A	N/A	0.3
HMM	2	1.3	0.4
CRF	2	3.6	0.3
MLN	5	82.0	47.0

4.5 Protocol

We evaluated the PGMs in a limited cross-validation study by first using Set #1 for training and Set #2 for testing, and then swapping the training and test sets. (We will conduct a more comprehensive CV study in the future, which will require careful separation of attack tracks into folds.)

We included two baseline algorithms: *Default* predicts that every instance is an attack, while *Perimeter Rule* mimics the perimeter defense strategy described in §2; it uses one feature per track instance (*Distance to HVU*).

For the PGMs, we tested window sizes from 1 to 100 in intervals of 5 (and a size of 2) and selected the size that maximized F_1. For MLNs, we started at a window size of 2 due to implementation constraints. Computation time constraints prevented us from reporting MLN results for window sizes of more than 60. For *Perimeter Rule*, we varied the triggering distance (between the HVU and the vessel being assessed) from 1 to 1000 meters.

4.6 Results and analysis

Table 4 displays the results for detecting attacks, and provides informal support for our hypothesis. As expected, *Default* performs poorly with respect to precision. *Perimeter Rule* performs well for the first set, where it learns a higher distance threshold, but not for the second, which involves testing on many more tracks of non-attacking vessels. MLNs attain the highest F_1 scores for both sets (e.g., 0.59 vs. 0.19 for the CRF model on Set #1), which may reflect their ability to better represent domain knowledge and learn weight settings from a few training instances. However, all PGM precisions decrease on Set #2, which we conjecture is because attack instances are rarer in Set #2 than in Set #1.

Table 5 displays the optimal window sizes found for the PGMs on one test and their corresponding training and test times (summed over all 320 time steps). MLNs better exploit temporal information in this domain than do HMMs or CRFs (which selected window sizes of 5, 2, and 2, respectively). MLNs require longer training times (82 seconds vs. 1.3 and 3.6 for HMMs and CRFs, respectively) and inference times (an average of 47/2310 = 0.02 seconds per track instance), although this is not excessive at this window size. However, while the HMM and CRF models record a linear increase in the time required for inference as window size increases, the time required to test MLN models increases exponentially. Therefore, if large window sizes are required to optimize MLN performance on this real-time task, then further research would be needed to increase the speed of applying MLNs.

5. Discussion

Applying PGMs to real maritime data was challenging for a variety of reasons, some of which we describe below.

5.1 Task-specific challenges

Detecting small-vessel threats from tracks obtained from USSV sensors poses several challenges. For example, our data is noisy and requires substantial transformation for use by the PGMs. It also includes tracks from many civilian vessels, which complicates this task but makes it more realistic. As mentioned, threat detection needs to be performed in real time, which poses challenges for some types of PGMs (e.g., MLNs). Threats should be detected as early as possible; we will address this metric in the future. Finally, our future work will also include assessing the abilities of PGMs to identify *coordinated* attacks from multiple vessels.

5.2 Modeling maritime threat prediction tasks

Like Crane and McDowell (2012) we found that considerable trial and error was required to apply MLNs to our task. It is difficult to isolate the effect of a specific rule on Alchemy's performance. We modified rules several times and even attempted to manually adjust the weights. However, this resulted in marginal improvement. To obtain more insight, we developed a result visualizer (similar to our annotation tool) that displays the learned activities over time. Using this tool, we identified the state transitions where these models perform poorly, which allowed us to more effectively adjust the features and rules.

The MLNs were highly sensitive to the type of modeling rules. Complex formulation of domain knowledge led to an explosion of the state space and made weight learning impractical (e.g., requiring days to compute). We examined a variety of rule formulations for their effects on weight learning and inference. However, unlike Crane and McDowell (2012), we did not find substantial improvement by including unit clauses in our models.

We also explored the use of more complex rules such as changes in distance across time intervals. This added another 144 clauses and increased the training time to 5 hours and inference to 3.5 minutes, up from 82 seconds for training and 47 seconds for inference using the optimal MLN model. These more complex rules also proved challenging for Alchemy's weight-learning procedure and invariably decreased rule set performance.

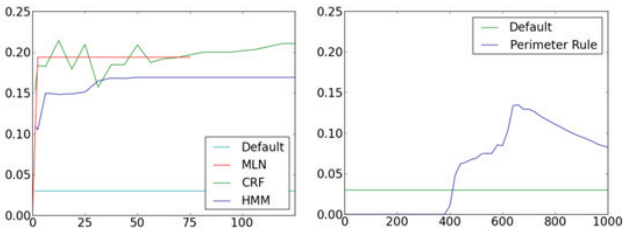

Figure 2: F_1 scores during training while varying (left) PGM window size and (right) *Rule* perimeter distance threshold.

5.3 Parameter tuning

Figure 2 displays, when training on Set #2, the F_1 scores from training the PGMs as the window size varies and the performance of *Perimeter Rule* as the distance threshold varies. For the most part, window size has only a minor effect on PGM training performance, and has little effect on the test performance. (In contrast, the distance threshold greatly affected *Perimeter Rule* training and test performance.)

6. Conclusion

Detecting small-vessel maritime threats is an important but challenging task. Deployed approaches, which use a perimeter defense trigger, are limited because they ignore vessels beyond this perimeter, as well as many of their features. We describe an initial application of three probabilistic graphical models (PGMs) to this task, and report some performance benefits.

However, many topics remain for future research. For example, efficient methods for MLN structure learning could greatly simplify our application of them to this task. Next, we have not yet addressed the topic of coordinated attacks, which could be represented using PGMs. Also, while we have studied knowledge-poor anomaly detection methods for maritime behavior recognition from local surveillance data (Auslander et al. 2011), and here study PGMs for threat detection, we plan to also study knowledge-intensive intent and plan recognition techniques for this task. Finally, we plan to test these techniques onboard unmanned sea surface vehicles under real-time conditions.

Acknowledgements

Thank to SIS for providing the data we used in our study, ONR for funding this research, and the reviewers for their encouraging comments and feedback.

References

Alchemy (2011). Alchemy – Open source AI. [http://alchemy.cs.washington.edu]

Auslander, B., Gupta, K.M., & Aha, D.W. (2011). Comparative evaluation of anomaly detection algorithms for local maritime video surveillance. In *Proceedings of the Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense X*. Orlando, FL: SPIE.

Bostwick, D., Goldstein, J., Stephenson, T., Stromsten, S., Tierno, J., Torrelli, M., & White, J. (2009). PARSEC, an application of probabilistic case based reasoning to maritime surveillance. In *Proceedings of the IEEE Conference on Technologies for Homeland Security*. Boston, MA: IEEE Press.

Brand, M., Oliver, N., & Pentland, A. (1997). Coupled hidden Markov models for complex action recognition. *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference* (pp. 994-999). San Juan, Puerto Rico: IEEE Press.

Crane, R., & McDowell, L.K. (2012). Investigating Markov logic networks for collective classification. In *Proceedings of the Fourth International Conference on Agents and Artificial Intelligence*. Vilamoura, Portugal: SciTePress.

Domingos, P., & Lowd, D. (2009). *Markov logic: An interface layer for AI.* Morgan & Claypool.

Gupta, K.M., Aha, D.W., & Moore, P. (2009). Case-based collective inference for maritime object classification. *Proceedings of the Eighth International Conference on Case-Based Reasoning* (pp. 443-449). Seattle, WA: Springer.

Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: Principles and techniques*. Cambridge, MA: MIT Press.

Lavee, G., Rivlin, E., & Rudzsky, M. (2009). Understanding video events: A survey of methods for automatic interpretation of semantic occurrences in video. *IEEE Transactions on Systems, Man, and Cybernetics*, **39**, 489-504.

Lipton, A.J., Heartwell, C.H., Haering. N., & Madden, D. (2002). Critical asset protection, perimeter monitoring, and threat detection using automated video surveillance. In *Proceedings of the Thirty-Sixth Annual International Carnahan Conference on Security Technology*. Atlantic City, NJ: IEEE Press.

Manfredotti, C. (2009). Modeling and inference with relational dynamic Bayesian networks. *Proceedings of the Twenty-Second Canadian Conference on Artificial Intelligence* (pp. 287-290). Kelowna, Canada: Springer.

Manning C. D., Raghavan, P, & Schütze H. (2008). *Introduction to information retrieval.* Cambridge, UK: Cambridge University Press.

McCallum, A.K. (2002). MALLET: A Machine Learning for LanguagE Toolkit. [http://mallet.cs.umass.edu]

Rabiner, L.R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, **77**(2), 257-286.

RemoteReality (2011). OmniAlert360. Retrieved on 20 November 2011 from [http://www remotereality.com].

Summer D. (2010). *Trident Warrior 10 data collection and analysis report for autonomous maritime navigation (AMN)*. Unpublished report.

Sutton, C., & McCallum, A. (2006). An introduction to conditional random fields for relational learning. In L. Getoor & B. Taskar (Eds.) *Introduction to Statistical Relational Learning*. Cambridge, MA: MIT Press.

Tran, S.D., & Davis, L.S. (2008). Event modeling and recognition using Markov logic networks. *Proceedings of the Tenth European Conference on Computer Vision* (pp. 610–623). Marseilles, France: Springer.

USV (2007). *The Navy Unmanned Surface Vehicle (USV) Master Plan*. [www navy.mil/navydata/technology/usvmppr.pdf]