Simultaneous Inference of Places, Activities, and Behavioral Classes in Maritime GPS Traces*

CASOS Technical Report

George B. Davis, Kathleen M. Carley

November, 2006
CMU-ISRI-07-113

Carnegie Mellon University
School of Computer Science
ISRI - Institute for Software Research International
CASOS - Center for Computational Analysis of Social and Organizational Systems

Abstract

Previous work has shown that activities and places of interest can be extracted from GPS traces of human movements using behavioral models based on conditional random fields (CRFs) [3]. In this paper, we adapt and extend this work in two ways. First, we apply the framework to analysis of a vehicle-tracking maritime environment, analyzing GPS data from a 5 day surveillance of merchant marine ships conducting exercises in the English channel. Secondly, we expand the model to a perform a broader population analysis segmenting the population into several classes with distinct behavioral models. Empirical results show that our algorithm is successful in inferring locations of interest, but makes only coarse distinction in activity inference. In clustering behaviors, it successfully divides agents with highly localized activities from those servicing distant ports.

Keywords: Machine learning, GPS, Graphical models

This material is based upon work supported by Office of Naval Research grant N00014-02-1-0973. Additional support at Carnegie Mellon University was provided by the Center for Computational Analysis of Social and Organizational Systems. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of the National Science Foundation, the Office of Naval Research, or the U.S. government.

Table of contents

I. Index of Tables

II. Index of Figures

1. Introduction

Localization systems such as the Global Positioning System (GPS) are making it possible to collect data in many domains tracking the position of individual actors over time. One way to use this data is to interpret it as the observed portion of a system which includes hidden contextual variables, such as the activities and goals motivating agent movement at each time period in the trace. If a distribution over values in such a system can be estimated, observed paths (and any other data collected simultaneously) can be used to make inference about hidden states. This framework has been successful in applications such as prediction of future motion [1], identification of transport habits and deviations therefrom [2], and annotation of traces with activities and activity-associated locations [3].

In each of the experiments above, a model was trained by fitting parameters to labeled traces from multiple users, then used to perform inference about new, unlabeled traces. Implicitly, this assumes that all traces (labeled or not) come from users sharing the same behavioral patterns. In many multi-agent environments this is incorrect: traces may come from agents of several classes with distinct behaviors. If we train a single model on all such traces, the resulting parameters would be fit to a mixture of behavior distributions, resulting in poor inference on any real trace. In this paper we address this possibility by first segmenting the labeled population into behavioral classes and training a separate behavioral model for each class. These models summarize patterns in labeled data and can be used to estimate the behavioral class of new traces, for better inference. We apply the technique to a set of traces tracking the movements of merchant marine vessels in the English channel. Potential applications in this setting include the isolation of distinct commercial strategies and distinction of “normal” commercial activity from deviant activities such as smuggling.

We accomplish the segmentation via supervised expectation-maximization (EM) clustering. Labeled traces (which correspond 1:1 to agents) are first divided randomly into K partitions. At each iteration, a behavioral model is trained for each partition based on the traces within. Each trace is then reassigned to the partition whose model gives it highest likelihood, and the two steps are repeated until partitions stabilized. To model behavior within each partition we adapted the approach of Liao *et al.* (2007), which associates each time segment in a trace with an activity and, for stationary activities, a place of interest. The conditional distribution of activities and places is modeled using conditional random fields (CRFs) generated for each trace. Training this model consists of learning weights on log-linear compatibility features between variables. At time of inference, places of interest are proposed and repeatedly updated in conjunction with identifying stationary activities in the trace.

The rest of this paper is organized as follows. In section 2, we provide background on our data and on conditional random fields. In section 3, we briefly summarize the use of CRFs for activity and place recognition, emphasizing the adaptations we made to previous work, and outline our EMapproach. In section 4 we present results for both single-model training and inference and our population segmentation task. Since this is a pilot study, these results are based on minimal labeled data and a simplified model, reducing their generalizability. Concluding in section 5, we discuss how future work can overcome these limitations.

2. Background

Maritime GPS Data

From the 25th to 30th of June 2005, a sensor network queried Automated Identification System (AIS) transponders on merchant marine vessels throughout the English Channel, recording navigational details such as current latitude and longitude, heading, speed, a navigational status such as “ANCHORED”, “MOORED”, or “UNDERWAY”, and several forms of identifying information. In total, traces for over 1700 vessels were recorded, with apparent activities ranging from simple shipping lane traversals to complex itineraries with stops at multiple ports of call.

A primary motivation in analyzing this data is distinguishing different navigation strategies that might affect port congestion or aide in differentiating normal trade behavior from “deviant” activities such as smuggling or logistics for terrorist activity. We also wanted to pick out locations around which these activities center, such as queues where ships wait for entrance to a busy port. Table 1 shows the range of activities we defined corresponding to those hidden values, as well as associated place types and the color code used in the figures in this paper.

Label	Meaning	Color Code	Associated Place
DOCKED	Transferring goods	Red	DOCK
WAITING	Waiting to enter	Pink	QUEUE
REFUELING	Open-sea refueling stations	Green	FUEL
LOCAL	Local maneuvering	Orange	(none)
TRANSIT	Unconstrained open-sea travel	Purple	(none)
LANE	Traversing shipping lane	Blue	(none)

Table 1: Activity and Place Labels

Because polling is scheduled independent of activity, our GPS traces contain many redundant entries (i.e. 5 days of hourly reports from the same dock). This unnecessary computational burden can be reduced by pre-segmenting the data into intervals and aggregating reports on each interval. Previous work, including [3], performed segmentation by “snapping” observations to street segments, under the assumption that an interval spent on a small segment was spent doing a single activity. Our ocean environment does not have a similar structure to snap to, and there is no satisfactory radius to contain an activity while reducing redundancy – shipping lane traversals might cover thousands of kilometers, but movement of even half a kilometer indicates that a ship is no longer docked. We experimented with several strategies and settled on segment boundaries determined by a significant change of speed (see section 3.1 for our discretization of velocity), since acceleration is the primary sign of control activity in these large vessels.

Segmentation was done automatically by a script which determined boundaries and then calculated aggregate statistics such as average speed and location for observations within a segment. After segmenting all data, we picked 10 long and varied traces to hand label based on domain knowledge. Figure 1 shows the range of this data, with pointer colors indicating activity labels for each segment (see table 1), and locations of interest marked with orange circles.

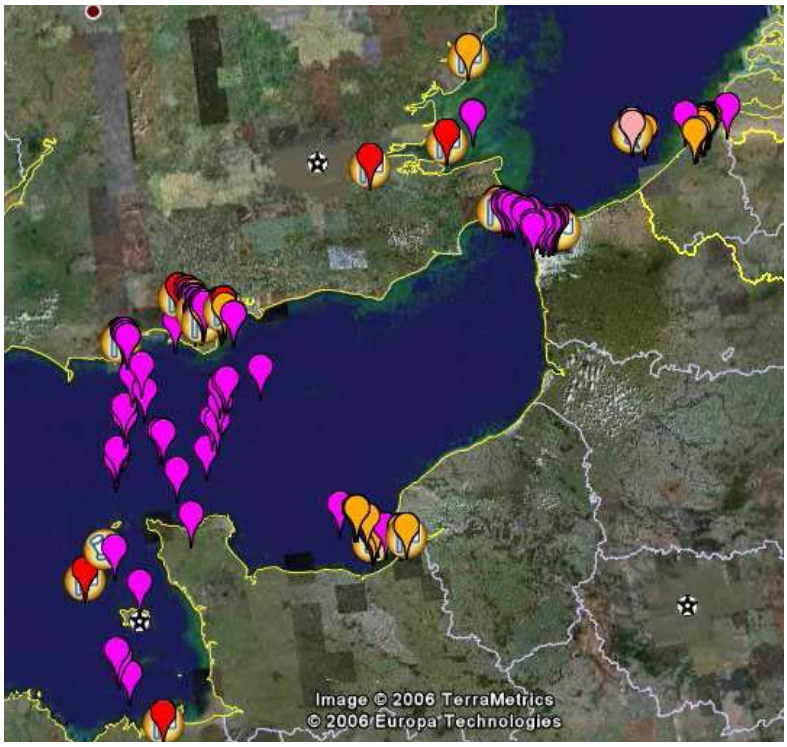

Figure 1. Labeled Segments and Places

Conditional Random Fields

A Conditional Random Field (CRF) is a graphical model whose maximal cliques C factorize a conditional distribution according to

$$P(Y \mid X) = \frac{1}{Z(X)} \prod_{c \in C} \phi_c(X_c, Y_c) \quad (1)$$

$$= \exp\left\{ -\log Z(X) + \sum_{f \in F} w_f f(X_f, Y_f) \right\} \quad (2)$$

where $Z(X)$ is a normalizing function, X_c and Y_c are members of clique c, and Φ_c is a corresponding potential function indicating the relative compatibility assignments to member variables. In equation (2), we have replaced the _s with a set of log-linear features, whose parameter sets will define the maximal cliques. If the weight of a feature is positive, then the feature is correlated with the compatibility of its parameter variables; negative-weights denote anticorrelation.

When we find model parameters for observed variables X and hidden variables Y, we are doing discriminative learning – we make no attempt to estimate any distribution over X. This is attractive because it allows us to estimate fewer parameters, and to be agnostic regarding independences between our X variables. On the downside, we cannot solve inference queries with unknown values in X. Defining our distribution in terms of features simplifies model design

and allows us to discretize continuous observed variables to avoid hybrid factors. It also gives a more interpretable parameterization, and one that is convenient for learning of shared parameters. On the downside, our model learning is constrained by the parameterization and cannot assign importance unanticipated interactions between variables.

3. Methods

3.1 Model

Figure 2 shows the layout of a CRF associated with a trace. There are 3 classes of variables:

- A set of observed variables **average speed** (S), **course variance** (C), and **navigation data** (N) are instantiated for each of the n trace segments, representing statistics aggregated over the observations within that segment. Average speed and course variance are real-valued, but navigation data is a binary vector indicating
- One **activity** (A) variable is instantiated for each segment with value range as listed in Table 1.
- A set of **place** (P) nodes are associated with only those activity nodes corresponding to time segments spent near them. Each place can takes on type values listed in Table 1.

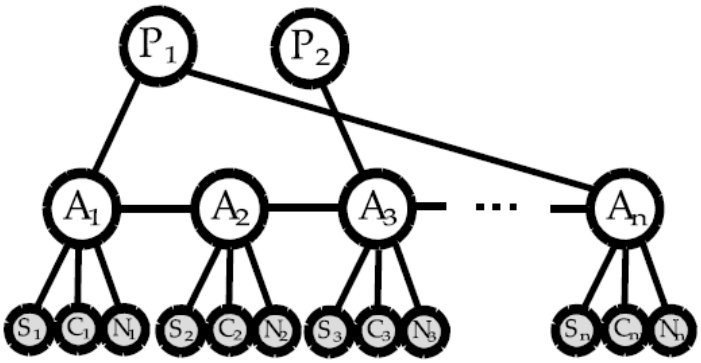

Figure 2. CRF Associated with a Trace

Note that actual location data is not directly encoded by variables in the model. Instead, the relevant aspect – association between places and activities – is encoded by the place ←→ activity edges. These are decided when the CRF is generated by testing whether each activity falls within a 5 km1 radius of each place, an assigning edges between nearby pairs. Locations near no place of interest are not connected to any place.

As a hypothetical example, the model in figure 2might correspond to a ship beginning with a DOCKED activity at DOCK P1, engaging in a TRANSIT activity in open ocean at A2, visiting FUEL station P3 and REFUELING at A3, before finally returning to P1 at time n.

The edges in figure 2 correspond to several classes of features:

- Place ←→activity edges represent sets of **compatibility** features. Each feature in the set corresponds to a specific pairing of place types and activities and is binary valued, returning 1 if that combination occurs. Positive weight on such a feature indicates that that activity often occurs at that type of place. Similarly,

- Activity←→activity edges are **transition** feature sets, which are set up like compatibility sets but for pairs of activities. Positive weights here indicate a oft-occurring transition.
- Binary **navigation** features are constructed similarly, but each one indicates the co-occurrence of one activity and a specific status message being observed in the segment time interval. Weights on these features capture the degree to which a status message such as "MOORED" correlate a specific activity such as "DOCKED".
- There is one **course variance** feature for each activity, which returns the real course variance value if the activity node is as specified. A positive weight here indicates that an activity involves frequent, large course corrections. The intention is to capture the difference between stable routes such as shipping lanes and more varied activity such as local maneuvering.
- **Speed** features operate like course variance features above, but there is an additional set of **speed class** features which discretize velocity. There is one such feature for each combination of activity and speed class. This is essential for 1) allowing activities to be associated with specific speeds rather than simply high or low speed, and 2) allowing us to model multimodal speed distributions. Figure 3 shows the distribution of observed velocities in our data as well as the breakpoints we selected between speed classes.

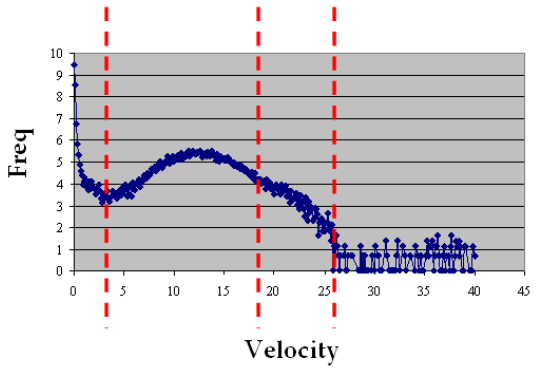

Figure 3. Velocity Histogram and Discretization

Weights are shared between all instances of the same feature in the network – for example, the weight on compatibility feature between place type "DOCK" and activity "DOCKED" is the same for any pair of place and activity nodes.

3.2 Parameterization

Training a model from labeled traces means learning weights for all of the features described in the previous section. To do this, we first construct a single CRF using all labeled traces (each trace forms a separate activity chain; traces are linked only when associated with a common place). We would now like to find an assignment of weights, W, that gives maximum likelihood to the labels known for this CRF. Liao *et al.* (2007) describe a fast approximate method called Maximum Pseudo-Likelihood (MPL) estimation, which we will briefly outline. The speed advantages of this method are significantly more important for our application, because our EM clustering will require multiple iterations of training as well as inference.

Maximum pseudolikelihood proceeds by maximizing $\prod_{y \in Y} P(y \mid MB(y_i), W)$,), where ***MB(y)*** is *y*'s Markov blanket (adjacent nodes), rather than full likelihood ***P(Y | X,W)***. This is much faster since we need iterate only over the value range ***R(y)*** of each individual *y*, rather than all possible combinations of assignments to these values. We calculate the pseudolikelihood gradient for the weight of one feature using

$$\Delta PL_f = \frac{W_f}{\sigma^2} \sum_{y \in Y} \left(-f(y \mid MB(y)) + \sum_{y' \in R(y)} P(y' \mid MB(y), W) f(y', MB(y)) \right) \quad (3)$$

In other words, the difference between an observed value of a feature and its expected value given the surrounding variables, plus a Gaussian prior pushing feature weights toward 0. The prior is essential to avoid overfitting small amounts of data, and in our case was necessary for numerical stability of weight values for features which never took nonzero values in our training data. We picked a Gaussian prior because it was convenient for directly manipulating the variance.

Previous work used quasi-newton methods to choose the rate of gradient descent, but implementation of this was outside the bounds of this project's time limit. We instead used an adaptive descent rate which would grow slowly while likelihood increased and shrink quickly if the maximal point was overshot. While this produced reasonable weights quickly, it was very slow to converge, and we were forced to cap our approximation at a fixed number of iterations.

3.3 Inference

To estimate (maximum a posteriori) the values of unobserved variables in our CRFs, we used Max-Product Belief Propagation, which is described for activity detection in [3] and with greater generality in [4]. The only interesting feature of our implementation was that feature functions were dynamically (bytecode) compiled to incorporate values of observed variables directly.

Liao *et al.* (2007) include in inference not only assigning values to known variables, but proposing the existence of place variables which are initially unknown. This is done by first conducting hidden-value inference on a placeless CRF, then creating place nodes for each segment whose belief location-associated activity. Since these places may have been based on bad initial activity estimates, the inference is repeated with the new graph, and the process iterates until the population of places is stable.

In that experiment, the aim was to discover places of significance to a single agent. Since we are interested in discovering places of interest to a whole class of agents, we consider inferred places as part of a learned behavioral model. We incorporate this into the previous place-detection algorithm simply by using a CRF containing known places in the initial step, and inferring additional places as necessary.

3.4 Clustering

Our primary extension to previous work it the application of an EM clustering algorithm to segment labeled data into separate population classes. The input to the algorithm is a set of labeled traces and a number of clusters k. The algorithm then proceeds as follows:

1. For each trace, choose one of the k clusters at random as an initial assignment.
2. For each cluster, train a model as in 3.2 based only on traces assigned to the cluster.
3. For each trace, calculate the *psuedolikelihood* of the trace under each model and assign it to model giving the highest score.
If any traces change assignment, return to step 2.

The use of pseudolikelihood rather than actual likelihood is necessary because calculating the normalization function for the CRF of a long trace would be intractable. In future work, we will investigate the possibility that the repetitive features of our model might allow a shortcut to computing the true normalization function.

4. Results

The model and algorithms discussed in section 3 were implemented in the Python interpreted programming language and executed on a 1.8 Ghz Pentium 4 laptop with 1 gb of RAM. We were able to produce only 10 labeled traces, for a total of 214 labeled segments and 15 places. We used these to conduct experiments in two phases. First, we performed leave-one-out validation of the model by comparing actual labels for each trace to those inferred using a model trained on the other 9. Then, we tested the success of our clustering technique in dividing the model into two populations.

Runtime. In our experiments, the most expensive procedure was model training. We capped the MPL algorithm at 20 iterations, which took an average of 253 seconds on datasets with 9 traces. MAP inference using Max-Product took an average of 12 seconds for a single unlabeled trace. The place detection algorithm never required more than 3 iterations to find a stable set of locations, with one MAP inference performed per iteration. The clustering algorithm took 4 iterations, each requiring 2 parameter learning phases. However, each training uses only a portion of the labeled population, and the total runtime was approximately 14 minutes.

Learned Weights. Our training process was successful in selecting weights which matched the intuitions on which we based our hand-labeling. For example, weights on the speed class features for the Docked state were negative except for a value of .011 for the lowest speed class. Features for the navigation status messages were carried near-zero weights, except that the "UNDERWAY" message carried weights near .01 in conjunction with nonstationary activities. Significant transition weights include a high probability of shifting from DOCKED to TRANSIT and high probabilities of shifting in either direction between TRANSIT and LOCAL features.

Leave-One-Out Accuracy. Our model had significant difficulties in differentiating all but the coarsest distinctions in utilities. Figure 4 shows that MAP inference assigns only DOCKED and TRANSIT activity labels, resulting in 64% training accuracy. On the simpler task of differentiating stationary from non-stationary activities, accuracy was 93.9%. Errors in this second task were caused mainly by unnecessary transitions between DOCKED and TRANSIT states, presumably from high transition probabilities and overlapping state compatibilities

between these states. As we discuss further in the conclusions, a potential solution to these issues is a larger amount of labeled data and a richer feature set.

Figure 4. Sample trace labeling.

Another question as regards inference is whether our model was able to accurately predict locations of interest to merchant marines. Although our algorithm did not produce precisely the same coordinates, there was a clear 1:1 correspondence for 14 out of 15 labeled places in our learned places. However, all learned places received the DOCK label, which is unsurprising given the inferred place labeling.

Clustering. Fortunately, our clustering algorithm does not depend on accurate inference, and was able to give interesting results even for our small set of labeled data. Figure 5 shows the two clusters identified in 4 iterations of EM. Qualitatively, they seem to differentiate behavior patterns with significant amounts of docking and local movement from those who involve large amounts of transit time.

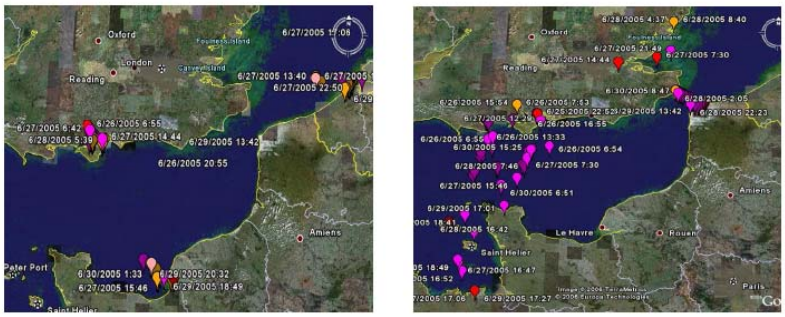

Figure 5. Clustering Results

5. Conclusions and Future Work

We adapted existing methods for extracting activities and locations of interest from GPS traces to a commercial shipping environment. We also extended that framework with a clustering step that segmented the population into groups with distinct behavior patterns. Our training and inference algorithms were unsuccessful in assigning precise activity labels to unlabeled traces

based on a small labeled data set, but was successful in identifying locations of interest. Our clustering algorithm was surprisingly successful in separating a small population of labeled traces into groups with clearly identifiable differences in behavior.

This pilot study could be improved into primary ways to make our results more generalizable. First, tests should be performed with substantially more labeled data. The few labeled traces we produced carried information about a small region of the conditional distribution For example, many activity transitions were never observed, resulting in clear overfitting on the weights of some features. Ideally, a larger set of traces would be labeled directly by ships crew or apparatus, rather than by third parties with domain knowledge.

Secondly, our model itself should be augmented with additional features that were too difficult to implement for the scope of this project. For example, features incorporating external information, such as known port locations or coordinate distance from land, could greatly improve the accuracy of our place type detection. Another useful addition would be counting and timer features to permit association of activities with durations and overall frequencies. These features would have to be designed carefully to avoid producing large maximal cliques in our model.

This framework for path analysis spans many fundamental problems, including graphical model training, inference, structural inference (with place nodes), and clustering. In each of these, we make compromises for performance: maximum pseudo-likelihood is an approximation, our MAP inference does not give a full view of the posterior, and both our place detection and clustering algorithms are susceptible to local maxima. While these compromises are necessary to make the model tractable, it is important that we characterize the error induced by these steps, both individually and in conjunction with each other.

References

[1] Daniel Ashbrook and Thad Starner. Using gps to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing*, 7(5), 2003.

[2] L. Liao, D. Fox, and H. Kautz. Learning and inferring transportation routines. In *Proceedings of AAAI*, 2004.

[3] L. Liao, D. Fox, and H. Kautz. Extracting places and activities from gps traces using hierarchical conditional random fields. *International Journal of Robotics Research*, 2007.

[4] J.S. Yedida, W.T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005.