Semi-Automatic Ontology Extension in the Maritime Domain

Gerben de Vries[a] Véronique Malaisé[b] Maarten van Someren[a]
Pieter Adriaans[a] Guus Schreiber[b]

[a]*Universiteit van Amsterdam, Human-Computer Studies Lab, Informatics Institute*
[b]*Vrije Universiteit Amsterdam, Business, Web and Multimedia Section, Department of Computer Science*

Abstract

One of the tasks of a maritime safety and security (MSS) system is to map incoming observations in the form of sensor data onto existing maritime domain knowledge. This domain knowledge is modeled in an ontology. The sensor data contains information on ship trajectories, labeled with ship types from this ontology. These ship types are broad and within one type there can be several distinctive behavior patterns. As a consequence we cannot make a good mapping from these trajectories to the ship types. To make this possible we should change the ontology by adding relevant subtypes. This paper presents a semi-automatic method to extend the ontology of ship types on the basis of trajectory data. The first part involves the use of hidden Markov models to model the data of each ship within one ship type and the clustering of these models. The clusters are input to the second part where we use internet querying and natural language processing based ontology extension techniques to extend the maritime domain ontology. We present the promising results of a preliminary experiment that shows an interesting possibility in terms of semi-automatic ontology extension, which would enable an optimal coverage of a given domain: not providing too many concepts, and not leaving essential ones out.

1 Introduction

The goal of maritime safety and security (MSS) systems is to provide their end-users with situational awareness: the ability to detect anomalies in maritime behavior, such as illegal fishing or tankers on a collision course. This is achieved by integrating information from different data sources and processing them intelligently.

One of the tasks in an MSS system is to map incoming observations in the form of sensor data-streams onto existing domain knowledge. Such sensor data-streams provide information about the movement of individual ships. The domain knowledge contains information about the static aspect of ships, it describes ship types and their corresponding characteristics.

More specifically, for the case described in this paper, we have data about ship trajectories on the one hand, and a formal description of ship types and their attributes, i.e. an ontology, on the other. This trajectory data is already labeled with ship types from the ontology, e.g. certain trajectories are from tankers, others from passenger ships, etc. However, these types are broad and do not distinguish all the different behaviors that we observe in the trajectory data. Within one type there can be several distinctive trajectory patterns that might belong to different subtypes. Thus, a concept (ship type) that describes a set of ships that is homogeneous in the existing ontological properties may cover ships with very different trajectory patterns. This means that we cannot classify ships on the basis of their trajectories, we cannot discover inconsistencies between trajectories and other information about ships and we cannot predict their trajectories from their type. In other words: the current ontology does not allow a good mapping between the sensor data and the existing domain knowledge. Therefore, the ontology should be changed.

How to discover possible subtypes on the basis of trajectory data and then to extend the domain ontology with these subtypes is the problem that we solve in this paper. We will show how to learn behavior models for individual ships, how to cluster these models and how to semi-automatically[1] extend the domain ontology on the basis of these clusters. The result is a combination of machine learning and ontology engineering that is novel and has interesting potential. Machine learning-wise the method is a nice approach to solving the classic unsupervised learning problem of labeling discovered clusters. Furthermore, we do semi-automatic ontology extension based on time-series (e.g. trajectories) data, which is a dynamic and untraditional data source.

The maritime safety and security domain is concerned with events as diverse as the rescuing of pleasure crafts, the monitoring of cargo ship routes or the prevention of smuggling, and objects as diverse as sensors (e.g. radar and sonar), ships, helicopters, off-shore petrol stations, etc. However, the data-set that we work with in this paper is more restricted, it comes from the automatic identification system (AIS). Every commercial ship larger than 300 tons is required to carry an AIS transponder. Such a transponder sends updates about, amongst others: position, heading, speed and rate of turn, every couple of seconds. Furthermore, the AIS specification defines different ship types, which is information that is also regularly communicated, hence our trajectories are labeled with ship types.

In the rest of this paper we first give an overview of the domain ontology that is created on the basis of the AIS specification. Then we will detail the main method in two parts. The first part is based on machine learning and finds possible subtypes by clustering hidden Markov models that model the individual ships' behavior. The second, ontology engineering part, is based on internet querying and natural language processing and, on the basis of the clusters of the first part, expands the domain ontology semi-automatically with new ship types. We will evaluate this method on the case of the 'Passenger Ship'. Finally, we end with a discussion, some conclusions and ideas for future research.

2 The AIS Ontology

To build the ontology modeling the AIS data information, henceforth the AIS ontology, we based ourselves on official documentation describing the different AIS messages: "IALA Guideline No. 1028 On The Automatic Identification System (AIS)"[2]. As the information was contained in tables of PDF documents and as there were very few concepts, with few defining constraints, we modeled this ontology manually, using a standard ontology editor: Protégé[3]. The ontology contains 141 classes and 21 properties, mostly related to the ship types' characteristics. There are only a few ship types: Tankers, Pleasure Crafts, Passenger Ships, High-Speed Craft (HSC), Wing in Grounds (WIG), Special Crafts (with 7 sub-types: Law enforcement vessels, Medical transports, Pilot vessel, Port tender, Search and rescue vessel, Tug, Vessels with anti-pollution facilities), and "Other Types of Ships". Their properties are linked to the fields in the AIS messages: their identifier, speed, length, draught, etc.

The classes of the ontology correspond to the different ship types that we want to distinguish in an MSS context. However, the ontology initially includes no information about trajectories. Taking into account trajectory information may motivate changes to the ontology: for example, a concept that describes a homogeneous set of ships in the original ontology (like Fishing Boat) may cover ships with very different trajectories (for example boats going to fish in the Atlantic ocean will behave differently from local fishermen). This makes it useful to split this concept into sub-concepts with different trajectory types. Therefore, it is mandatory to develop an ontology extension mechanism. As redesigning an ontology is a time-consuming operation, we developed a semi-automatic method, which we will describe in the next sections.

3 Learning and Clustering Ship Behavior

The goal of the method described in this section is to discover possible subtypes of a ship type defined in the AIS ontology on the basis of the trajectories of ships of that type. Trajectory data is an example of multidimensional time-series data. In this time-series data we want to discover clusters (to suggest possible subtypes). The most straight-forward thing to do would be to try to directly cluster the time-series. However, the time-series are very variable in length and they are multidimensional. Moreover, we know very little

[1]The full automatization of this process is still a work in progress at the time of writing of this paper. However, the results will always be validated by a human operator, therefore, the extension mechanism is qualified as semi-automatic.

[2]IALA is the International Association of Marine Aids to Navigation and Lighthouse Authorities.

[3]Protégé is a free, open source ontology editor and knowledge-base framework, available at `http://protege.stanford.edu/`.

about the behavior patterns that we are looking for. Therefore, we have no idea what a good distance measure would be to use for clustering the time-series. In such a case, it makes more sense to use a model based method to compare them, instead of comparing them directly (for a good overview on this, and other issues related to time-series, see [7, Chap. 5]).

The method described below contains three steps. First we preprocess the data, which reduces the size and complexity of the data and produces the required input for the second step. This second step is to train hidden Markov models as 'behavior' models for each individual ship. Thirdly, we define a distance measure on these models and use that to cluster them together with affinity propagation. Although the description of these steps is focused on the case of ship trajectories, the overall method is in principle applicable to any form of time-series data.

The AIS data-set contains trajectory data for a big number of ships, labeled with different types. Let us call a set of ships with the same type: $Ships$. For each $s \in Ships$ we use two time-series: rate of turn, $\lambda^s_{\mathrm{rot},1}, \dots, \lambda^s_{\mathrm{rot},n}$ and speed over ground, $\lambda^s_{\mathrm{sog},1}, \dots, \lambda^s_{\mathrm{sog},n}$. These time-series are all sampled at the same fixed sample-rate of ten seconds. We chose these two time-series because they describe the relative behavior of a ship, not the absolute behavior (e.g. this would be the case with position data).

3.1 Preprocessing

To reduce the size of the data-set, the time-series (per ship) are averaged with contiguous, non-overlapping windows of length w. This method is called piecewise aggregate approximation [5]. For a time-series $\lambda_1, \dots, \lambda_n$, let $\bar{\lambda}_1, \dots, \bar{\lambda}_m$ be the reduced version defined by[4]:

$$\bar{\lambda}_i = \frac{1}{w} \sum_{j=w(i-1)+1}^{w.i} \lambda_j \quad . \tag{1}$$

Experimentation has shown that $w = 30$ (300 seconds) strikes a good balance between capturing the general movement of ships and ignoring noise.

By making symbol strings representing the time-series, data is reduced even further. Also, it 'reduces' the multidimensional time-series to one dimension. Moreover, discrete data is the simplest input for hidden Markov models. The first step in this process is to create the set:

$$D = \{(\bar{\lambda}^s_{\mathrm{rot},i}, \bar{\lambda}^s_{\mathrm{sog},i}) | s \in \mathit{Ships} \wedge 1 \leq i \leq m\} \quad . \tag{2}$$

Then we make the set D_z by applying z-normalization (see for instance [2, Chap . 9]) to D, i.e. for both attributes we subtract the mean (μ) from each data-point and then divide each data-point by the standard deviation (σ).

The next step is doing standard k-means clustering (see again [2, Chap . 9]) on D_z to get clusters C, i.e.

$$C = \textit{k-means}(D_z, k) \quad . \tag{3}$$

We use the common Euclidean distance as distance measure. For k we chose a value that worked well in practice: $k = 5$. Each cluster $c \in C$ is assigned a symbol a_c, thereby creating a vector code-book. With this code-book we create a symbol string for each $s \in \mathit{Ships}$ by assigning the symbol $a_{c'}$ to $p_i = (\bar{\lambda}^s_{\mathrm{rot},i}, \bar{\lambda}^s_{\mathrm{sog},i})$, such that

$$\neg \exists c \in C : dist(c, p_i) < dist(c', p_i) \quad . \tag{4}$$

Where $dist$ is the distance measure used in k-means clustering, the Euclidean distance in this case, and p is z-normalized with the μ's and σ's that we used for D_z. Let Ships_{SYM} be the total set of symbol strings, one for each ship.

3.2 Hidden Markov Models

To capture the general behavior of a ship we use a hidden Markov model (HMM) [8]. These models are tried and tested and often used for modeling sequential data, therefore we chose to use them. They are more powerful than regular Markov chains, but less powerful than newer methods, for example conditional

[4]For notational ease we ignore the case when the length of the time-series is not neatly dividable by w and we are thus left with an incomplete last window.

random fields. However, one could use other models than HMMs in this step, provided that one can define a distance measure on such models, so that they can be clustered.

We train multiple HMMs $\gamma_{s,1}, \ldots, \gamma_{s,k}$, where $\gamma_{s,i} = (T_{s,i}, E_{s,i}, \pi_{s,i})$, on a set random subsequences Λ_s of each $s \in \mathit{Ships}_{SYM}$, using the standard EM (for HMMs also called Baum-Welch) algorithm. The transition matrix $T_{s,i}$, emission matrix $E_{s,i}$ and initial state distribution $\pi_{s,i}$ are all initialized randomly. The model i, such that

$$\neg\exists\gamma_{s,j} : j \neq i \wedge \sum_{\lambda\in\Lambda_s} P(\lambda|\gamma_{s,j}) > \sum_{\lambda\in\Lambda_s} P(\lambda|\gamma_{s,i}) \tag{5}$$

is selected as the final model for the ship s.

Using a test selection of ships we have identified that the ‘elbow’ point in the performance increase[5] gained by increasing the number of states in the HMM is around 6 states, with $k = 5$. Depending on the length of the behavior that is relevant, different length subsequences should be used, e.g. to learn a behavioral pattern that is 1 hour in length, one should use subsequences that are significantly longer. We used a length of 6 hours for our experiments. The total length of the number of subsequences can be in the same order as the length of the original sequence, to make sure that training does not take more time than necessary.

3.3 Clustering

To cluster hidden Markov models a distance/dissimilarity measure is required. Rabiner defines a rather simple one in his classic paper [8][6]:

$$Dis_{HMM}(\gamma_1, \gamma_2) = \frac{1}{n}(\log P(\lambda|\gamma_1) - \log P(\lambda|\gamma_2)) \ . \tag{6}$$

Where $\lambda = \lambda_1, \ldots, \lambda_n$ is a sequence generated by model γ_2. To make it usable with standard clustering algorithms this measure can be made symmetric:

$$Dis'_{HMM}(\gamma_1, \gamma_2) = \frac{Dis_{HMM}(\gamma_1, \gamma_2) + Dis_{HMM}(\gamma_2, \gamma_1)}{2} \ . \tag{7}$$

In our experiment we actually take a set of sequences instead of a single one. Furthermore, this set is the same as the set of sequences that we used to train the models. So, the measure that we use is biased towards the data that the models are trained on. We chose to do it in this way simply because the results are better.

Using the above measure we can compute a dissimilarity matrix for our set of individual ship models. This dissimilarity matrix can then be input to a clustering algorithm to find clusters of models. As a clustering algorithm we took the relatively new, and very fast affinity propagation (AP) [3]. It works by treating data-points as nodes in a network that pass (voting) messages around, thereby determining in a number of iterations what the best exemplars in the network are. Each data-point is linked to an exemplar, thus we have clusters with exemplars as their prototypes. The algorithm can deal with almost all kinds of distance/(dis)similarity measures (even non symmetric ones). It has two parameters to set: the ‘dampening factor’, which controls the rate of convergence, and the ‘preference’, which influences the number of clusters that are discovered.

Traditional hierarchical clustering can be used in this step as well and should provide similar results. The output of the above clustering step is input to the ontology extension process, which is described in the next section.

4 Extending the AIS Ontology

The previous step generates ship clusters, within the types from the AIS ontology. Thus, behavior-based clustering provides more fine grained ship types than the ones already present in the ontology. We have to extend the ontology to provide relevant label(s) for the unlabeled class(es). The methodology that we have defined is to extract the possible labels from Internet, based on the population of the unlabeled clusters: we take as input one field of information in the AIS message, the international identifier of a ship (its MMSI[7])

[5] The performance is measured by the probability of generating the training sequences.

[6] This is essentially an approximation of the Kullback-Leibler divergence, although Rabiner does not mention it there explicitly.

[7] MMSI stands for Maritime Mobile Service Identity.

and query automatically online AIS databases[8] providing ship types amongst other information. We then process the resulting pages and extract all possible ship types per cluster. We use the number of times a specific type is mentioned as an indicator of correctness and discard the types occurring only once, because of the noisiness of the data: sometimes MMSI numbers are incorrect and the databases can also contain erroneous information. Once the label is proposed, the corresponding concept has to be integrated at the right place(s) in the ontology. Our semi-automatic ontology expansion mechanism provides only suggestions, to be validated by a domain expert, but might provide more than one possibility. Its full automatization is still work in progress.

It consists of two steps: (1) querying the internet for explicit relationships between (a set of) new concept(s) and (a specific set of) the ones from the ontology and (2) extracting from the resulting files the pairs of concepts for which the queries gave at least one hit, with their related number of hits, and propose hierarchies of concepts based on: the extracted pairs, their number of occurrences, the existing hierarchy and information about the general domain. The resulting proposals for extending the ontology can then be validated or modified by the domain expert. These processes can be compared to the general principles of corpus-based ontology building, like in [6] or [1]: finding terms in textual resources, mining relationships between them, proposing these to domain experts and formalizing the validated results. But our method differs on two points: (1) we base ourselves on a "seed ontology" and intend to find only extensions to the core domain model in the textual resources and (2) we do not rely on a manually selected set of texts from the domain, that might be limited or biased by an expert's selection. Mining the relationships between concepts from the internet also implies a bias, but provides a certain objectivity (at least it goes beyond the selection made by a limited number of experts) and coverage of the domain. The two steps of our methodology are detailed below.

4.1 Querying the Internet to Define New Concepts

Search engines like Google or Yahoo! provide a search API[9] to automatically run a set of queries on their large indexes. We generate an automatic queries list with a specific format: besides the fact that they contain one lexical representation of a concept from the existing ontology and one candidate label of the new concept(s), these two also appear in the query in a standard "Hearst pattern". These patterns have been defined in the field of natural language processing for extracting subclasses from plain text corpora [4]. They contain lexical or semantic elements that can be tailored to a specific domain. We used the most productive and the most domain independent of these patterns: "*(X) such as (SuperClassOfX)*" and "*(X) and other (SuperClassOfX)*", where *X* is the new concept and *SuperClassOfX* is a possible superclass. As these patterns require also the plural form of the label of the classes, we used the Celex lexicon[10] to compute these, and built all possible queries (queries including the singular or plural form of all the concept's labels concerned, in both of these patterns). So, in our case *X* represents any textual representation of the new concept and *SuperClassOfX* any textual form of the different ship types already present in the AIS ontology, which would then correspond to a superclass suggestion for the new concept.

4.2 Proposing New Hierarchies

The pairs of terms for which the queries provided at least one hit are extracted from the results' file (so the new concept and another ship type), along with their number of hits (corresponding to their number of occurrences on the internet), as a measure of relevance for the relationship between the concepts of the pair. This number, the analysis of the existing hierarchies in the AIS ontology (the fact that some ship types are already modeled as subclasses of each other) and some domain knowledge (like the fact that a Ship and a Craft can be considered as subclasses, according to Merriam Webster's[11] online definition: "Ship: a large craft for travel by water") are used to propose possible hierarchies for extending the ontology, and rank the propositions. The ranking procedure is not automatized yet, but follows these heuristics: rank the pairs of terms based on the number of hits of the Hearst patterns corresponding to their lexical representations, check the existing hierarchy in the ontology to find subclasses within the proposed possible super-classes of the

[8]We queried the Google database at `http://aprs.fi/info/` and the one from the International Telecommunication Union at `http://www.itu.int/cgi-bin/htsh/mars/ship_detail.sh?`.

[9]`http://code.google.com/apis/ajaxsearch/documentation/` and `http://developer.yahoo.com/search/`

[10]`http://www.ru.nl/celex/`.

[11]`http://www.merriam-webster.com/thesaurus/ship`

new concept, use domain knowledge to find synonyms or further sub-classes between them and propose the resulting hierarchies' leaves as ontology extension suggestions.

5 Experiment

To test the above method we have set up an experiment using the AIS data-set.

5.1 Clustering Passenger Ships

In the data it occurs a number of times that ships stop sending AIS signals, for instance because they are in port or leave the area of coverage. If the time difference between the last signal and the new signal is significantly long (e.g. fifteen minutes), then we treat this signal as if it is the start of a new sequence, i.e. one ship can have a *set* of time-series.

From the AIS data we have selected all the ships that were labeled 'Passenger Ship' according to the AIS ontology. This data was first preprocessed as described earlier. For each ship we trained 10 hidden Markov models, with 6 states. We then selected the one with highest performance, such as defined in (5), for each ship. Using the dissimilarity measure (7) we computed a dissimilarity matrix as input for clustering with affinity propagation (AP). One of the parameters of AP, the 'preference', depends on the maximum dissimilarity in the dissimilarity matrix, and it requires some tuning by hand. In this experiment -12 worked fine, but unfortunately, as said, it completely depends on the dissimilarity matrix. The 'dampening factor' is less crucial and is fine on the standard setting of 0.95.

The result was two clusters. When we visually inspected the trajectories of the ships in the different clusters we saw that they were clearly distinct. One cluster showed the pattern of ships moving between two harbors regularly, with short intervals. The other cluster contained trajectories that had ships sailing in long, more or less straight, lines. The contents of these clusters are passed on to the next step.

5.2 Extending the AIS Ontology

5.2.1 Mining possible labels from the internet

As mentioned previously, this operation is performed in two steps: searching the internet for possible labels for the two ships' clusters and processing the resulting pages to isolate this precise information: the labels. We queried the aforementioned AIS databases with the MMSIs of the ships from the two different clusters and processed the resulting pages (fetched and stored locally): we extracted the ship types that were mentioned for each successful query.

Results for cluster number one This cluster contains 12 ships, for 9 of them the databases gave a type result: Passenger Ship "general" (4 times), Passenger Ship carrying hazard/pollutant cat A (3 times), Passenger Ship carrying hazard/pollutant cat C (once) and Ferry boat (once). Given the fact that we want to cluster ship types based on their behavior, no relevant sub-class of Passenger Ship should be modeled based on the first three types: a Passenger Ship carrying a specific cargo is dealt with as a property attached to the generic Passenger Ship class, this specification does not require a specific subclass in the ontology. The concept of Ferry, however, has a very specific behavior amongst the Passenger Ships and this concept is not present in the ontology yet. This would be a necessary extension to the ontology, if the Ferry label would represent a (whole and) separate cluster. However, the behavior of this particular ship was not distinguished from the ones of the generic Passenger Ship type by the machine learning process. Given the fact that there is also only one single instance of this specific type, in an otherwise coherent cluster, and given the fact that real-life data is often noisy, we did not create the extension for this single case.

Results for cluster number two We got results for 5 out of the 6 ships of the second cluster, but for 3 ships the information given by one of the databases was merely "Not available". For the same 3 cases, the "Ship Class" provided by the other database was "MM FBT", which means Merchant Ship - Ferry Boat. For one of the other 2 ships, the MMSI number was erroneous (a default number), and the last one was classified only as "Passenger Ship 'general"'. We decided to use the information given by 3 of the 6 ship's identifiers, and consider that the second cluster represented Ferries. An observation of the trajectories of the corresponding ships, and the additional knowledge about their starting and end points (ports linked together by a Ferry line) confirmed the validity of this intuition. We consider the one labeled "Passenger Ship 'general"' to be noise.

We had a closer look at the Ferry from the first cluster, to get insight about why it had not been clustered with the others. It turned out that this ferryboat was linking harbors from two different countries; our AIS data concerns only the information sent in one of these two national waters, thus we had only partial information about that ship's trajectory: the messages were stopped when the ship entered the second national waters, and we could not detect the typical "ferrying" pattern.

Once the Ferry's label is elicited, we wanted to have the corresponding concept semi-automatically integrated in the ontology, in order to update our domain knowledge. The method that we have employed for this is presented in the next section.

5.2.2 Analyzing the results and updating the ontology

We created automatically (by Perl scripts) a list of queries for the Yahoo! Search API, and stored the XML results locally. We processed the results and extracted from the files:

- the pairs of terms in a sub-class relationships for which the query had provided hits,
- the number of possible results (the number of hits that Yahoo! found in its index) and
- the number of retrieved results (as common internet users do not usually go beyond the third result page, we had set as threshold to retrieve at most 30 hits, so the actual number of pages is comprised between 0 and 30).

We got 2309 potential results (number of hits on Yahoo!'s index of the internet) and 262 actual ones (really retrieved pages, according to our threshold of 30 pages at maximum). The couples of concepts from the AIS ontology that were found on the Internet in a superclass relationship with the concept of Ferry were: Ships (2085 possible hits), Crafts (171), Passenger Ships (29), Other Types Of Ships (15) and Pleasure Crafts (7). Tankers and Fishing boats were not returned by our methodology, although the latter is often in close co-occurrence with one of the possible labels of the concept of Ferry on Internet. This shows the advantage of using our method over using a similarity based on simple co-occurrences in the internet in general (a principle that inspires, for example, the Google distance score between two terms): it gives more precise results. We intend to create hierarchies as specific as possible. Lexical inclusion indices, as investigated in [9], show us that Pleasure Craft have good chances to be a subclass of Crafts because "Crafts" is included in the label "Pleasure Crafts" and because the grammatical category (or Part Of Speech (POS) tag) of the extension is an adjective. In the same fashion, Passenger Ships could be a subclass of Ships, but with lesser certainty because this time the POS tag of the extension is a noun.

Nevertheless, as a first approximation, we propose the following hierarchies: Ships $>$ Passengers Ships $>$ Ferries, Crafts $>$ Pleasure Crafts $>$ Ferries and Other Types Of Ships $>$ Ferries. Pleasure Crafts, Passenger Ships and Other Types of Ships are already present in the ontology, as subclasses of Ship Types, so we suggest only the last level of these hierarchies as extensions. Ferries, according to what we found on Internet and to the core domain knowledge that we had, can either be a Passenger Ship, a Pleasure Craft or an Other Type of Ship. Considering the number of hits per possibility, and the fact that all the ships from the second cluster are all initially labeled with the broad category Passenger Ship, we give a higher rank to this suggestion. This suggestion has, however, to be validated by a domain expert, who can choose to keep all three possibilities, just two or decide for another one.

We still have to automatize this ranking and develop an interface for the interaction with the domain expert to update the AIS ontology directly. This automatization is currently in progress.

6 Discussion, Conclusion and Future Work

We have presented a method to semi-automatically extend a maritime domain ontology on the basis of trajectory data. It discovers possible subtypes of an existing ship type by finding clusters of behavioral patterns in trajectories for that type. Based on these clusters we use internet querying and natural language processing techniques to discover labels, and their place in the ontology, for the suggested subtypes.

From the perspective of machine learning it is a nice approach to solving the ever existing unsupervised learning problem of labeling discovered clusters. In ontology engineering the idea of semi-automatic ontology extension from the point of time-series data is rather novel. The presented method holds promise for the future: the combination of machine learning and ontology engineering works well and this application shows that there are interesting possibilities to be explored.

However, it is clear that more experimental evaluation is required. Not only do we need to do this in the maritime domain, but it would also be useful to test the methods on other comparable domains, the first that come to mind are other domains with moving objects, such as cars and planes. In principle the methods can be applied to any domain with dynamic, time-series data on the one hand and static, ontological knowledge on the other hand.

Another problem is that, for now, we need good information about the specific instances of the domain to be available on the internet. It is a challenge to find other properties of the clusters (i.e. not the specific instances) that we can use for the internet search.

Also, it would be nice if clustering of the entire data-set could be the basis of the ontology, instead of using a hand-crafted "seed"-ontology as we do currently (e.g. this would be useful in the case of completely unlabeled trajectory data). However, at least with the AIS set, the current approach is not able to do that well. For one thing because it is very slow, since it requires the computation of a large dissimilarity matrix. Furthermore, the currently used AIS attributes (rate of turn and speed over ground) do not seem to provide enough discriminatory information.

The hidden Markov model as the basis for behavior models was chosen because it is well-known and often used, however, this is not to say that there might not be models that represent behavior more accurately. Furthermore, the current training of the HMMs, based on random initialization, is rather slow, for other models we might be able to this faster.

Apart from these issues, we have more ideas for future work. First of all, we will look at using more data sources: other kinds of sensor data streams (for instance radar), but also other types of sources of domain knowledge. Currently there is no feedback from the extended ontology back in to the machine learning. Potentially this can be very interesting, because with the new labels we can start the process again and extend the ontology even further. An interesting issue here would be when to stop this loop to get optimal coverage of the domain: not providing too many concepts, and not leaving essential ones out.

References

[1] Brigitte Bibow and Sylvie Szulman. *Knowledge Acquisition, Modeling and Management*, volume 1621/1999 of *Lecture Notes in Computer Science*, chapter TERMINAE: A Linguistics-Based Tool for the Building of a Domain Ontology, pages 49–66. Springer Berlin / Heidelberg, 1999. http://www.springerlink.com/content/8e7p79nh40y3pbt6/.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[4] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, pages 539–545, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[5] Eamonn J. Keogh, Kaushik Chakrabarti, Sharad Mehrotra, and Michael J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *SIGMOD Conference*, 2001.

[6] Alexander Maedche and Steffen Staab. Kaon - the karlsruhe ontology and semantic web meta project. *KI*, 17(3):27–, 2003.

[7] Fabian Mörchen. *Time Series Knowledge Mining*. PhD thesis, Philipps-University Marburg, 2006.

[8] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[9] Eric SanJuan, James Dowdall, Fidelia Ibekwe-Sanjuan, and Fabio Rinaldi. A symbolic approach to automatic multiword term structuring. *Computer Speech & Language*, 19(4):524–542, 2005.