Learning Symbolic Descriptions of Activities from Examples in WAAS

Jongmoo Choi and Gérard Medioni
Institute for Robotics and Intelligent Systems
University of Southern California, USA
{jongmooc,medioni}@usc.edu

ABSTRACT

We present an automatic system that learns symbolic representations of activities from examples in Wide Area Aerial Surveillance (WAAS). In the previous work, we presented an ERM (Entity Relationship Models)-based activity recognition system in which finding an activity is equivalent to sending a query, defined by SQL statements, to a Relational DataBase Management System (RDBMS). The system enables us to identify spatial and geo-spatial activities in WAAS as long as activities are carefully defined by human operators. Here, we show how to infer a structured definition of an activity from examples provided by a user. Our system randomly generates a set of possible SQL statements using a logic generator in a MCMC framework, uses a memory-based RDBMS to validate generated SQL statements with the input data/database, and selects the best answer that allows the RDBMS to explain the input positive examples while excluding negative examples. We have evaluated our system on real visual tracks. Our system can find activity definitions from input examples and associated query results including motion patterns (e.g., "loop") and geospatial activities (e.g., "parking in a lot").

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications - *Spatial database and GIS*

General Terms

Algorithms, Experimentation

Keywords

wide area aerial surveillance, activity recognition, activity recognition in WAAS, rule and pattern mining, SQL from data

1. INTRODUCTION

We aim to provide a system that automatically finds symbolic definitions of activities given examples in WAAS (Wide Area Aerial Surveillance) scenarios.

In the previous research, we have developed EAR (Entity relationship models-based Activity Recognition) framework [Choi et al. 2012]. We extract a set of atomic portions of a track ('tracklets') from video input, along with physical attributes, and store them into a standard RDBMS (Relational Database Management System). To infer activities, we define relationships between the database entities, which capture the geo-spatial relationships between tracklets and and geographic information. The key limitation of this approach is that each activity needs to be carefully crafted and expressed by a human operator, which limits the efficiency and applicability.

In this paper, we present an automatic system that learns symbolic representations of activities from examples in Wide Area Aerial video Surveillance (WAAS). We do not aim to build a universal learning framework but instead leverage experts domain knowledge on WAAS applications. Instead of putting implicit knowledge on the final stage (activity definition), we design meaningful, efficient building blocks enabling to produce complex activity definitions. We then generate a set of activity representations and find the best one that explains input data.

Given positive and negative examples, our system randomly generates a set of possible SQL statements using a logic generator in a MCMC (Markov Chain Monte Carlo) framework [Gilks 2005], uses a memory-based RDBMS to validate a generated SQL with few example data, and selects the best answer that allows the RDBMS to produce the input positive examples while not producing negative examples.

We have implemented our framework and validated the approach on real visual tracks and GPS datasets. In our experiments, our system allows us to find motion patterns and geospatial activities from real visual tracks, with very few example data as input.

2. RELATED WORK

We summarize related work on rule mining from data. Related work on activity recognition and activity recognition in wide area surveillance can be found in [Choi et al. 2012].

Association rule learning is a well researched method for discovering interesting relations between variables in large databases [Agrawal et al. 1993]. As opposed to association rule learning applications, our problem has complex activity

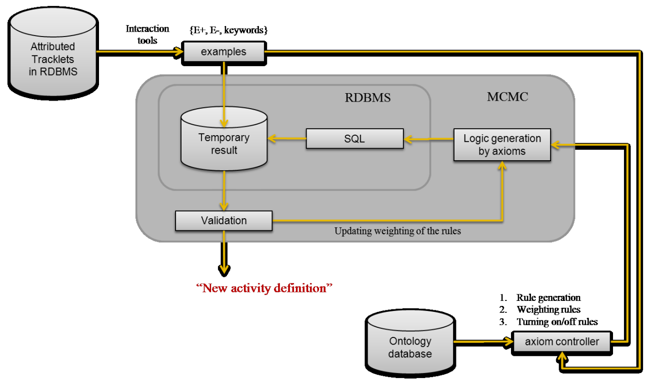

Figure 1: Overview of the proposed approach.

patterns (e.g., geographic context, sequence, multiple actors) and a limited number of positive examples.

Generating rules from unstructured raw data is a difficult problem. To make the problem more tractable, many works have focused on finding operators while a background knowledge setting is kept. A hybrid approach combining AI planning and evolutionary optimization, within an interactive storytelling framework, is presented to generate planning operators [Giannatos et al. 2012]. Rule extraction methods from trained neural networks have been proposed [Chorowski and Zurada 2011].

In our approach, the problem of extracting rules from data is cast as a search problem with operators, input examples, and background data. Our approach is efficient and practical for WAAS applications because of the utilized domain knowledge. Moreover, this framework can be extended by replacing a domain specific set of rewriting rules.

3. PRELIMINARIES

We summarize methods for tracklets computation and geographic object extraction. We then explain the basic concept of EAR framework.

Computing tracklets from imagery. The atomic spatio-temporal information in our system is called a tracklet, a segmented portion of a track representing vehicle's "instantaneous" motion, such as going straight, turning left or turning right. Each tracklet has a collection of attributes $x_i = \{\lambda_1, \lambda_2, \cdots, \lambda_m\}$, where an element λ_i presents a physical property such as time, location, and speed. We use a state of the art real-time tracker [Prokaj et al. 2011], which has been used by Lawrence Livermore National Lab.

Extracting geographic objects. Given a ROI (Region Of Interest) for an activity inference application, we extract a set of geographic objects (e.g., buildings, roads, parking lots, schools) from OpenStreetMap [Haklay and Weber 2008] and store into our RDBMS along with key attributes (e.g., name, id, type, latitude, longitude). A road is represented a set of road segments and a large object, having an arbitrary shape of the area (e.g., a parking lot), is represented as a set of points. All geo-objects are stored and indexed into the RDBMS by an off-line processing step.

EAR (ERM-based Activity Recognition. In the EAR framework, an activity a_j is defined as a collection of tracklets obeying certain properties [Choi et al. 2012]: $a_j = \{x | x \in \Omega_j, C_j(x) > \theta_j\}$, where Ω_j, $C_j(x) \in [0, 1]$, and θ_j represent the relationship associated with the activity, the confidence function and the recognition threshold, respectively. The relationship Ω_j links between the attributes of entities, which include both the physical properties of tracklets and the geospatial data. We can define relationships that are not explicitly represented in the ERM. Some examples can be found in [Choi et al. 2012].

The ERM-based representation implies that inferring an activity is a search problem to find a subset of tracklets from entire data set, which satisfies certain conditions. ERM is implemented as a standard RDBMS and we can express set operations by SQL to find an activity from our database. The activity recognition problem is equivalent to sending queries to the RDBMS.

4. OUR APPROACH: MCMC-LOGIC SAMPLING FROM AXIOMS

We propose an innovative framework that allows us to find activity definitions without relying on experts' explicit SQL design. The input to our system is a few positive and negative examples of tracked vehicles. An optional natural language description can be provided. Fig. 1 shows an overview of the system. The system randomly generates a set of possible SQL statements using a logic generator in a MCMC framework, uses a memory-based RDBMS to validate a generated SQL with few example data, and select the best answer that allows the RDBMS to produce the input positive examples while not producing negative examples.

The key idea is that we randomly generate logic representations (SQL statements corresponding to the ERM) using a set of axioms and select the best representation that fits to the input examples using a series of DBMS queries. In general, this bottom-up search is a combinatorial problem and intractable. Our rewriting rules allow us to generate only feasible logic statements which are consistent with the input data.

A simple ontology database can be used to reduce the search space, which is not presented in this paper.

4.1 Logic generation from axioms

We use a set of rewriting rules (axioms) to generate a set of logic statements and convert each logic representation to a corresponding SQL statement that can be evaluated by the RDBMS. Each activity is represented by SQL statements in this ERM-based activity recognition framework. A SQL statement has a WHERE clause that represents a set of properties to define the activity. We represent the WHERE clause using a binary tree. Each node in the tree has a pair of sub-expressions and/or values. We define a set of rewriting rules:

$$\begin{aligned} e &\to \{e, \cap, e\} \\ e &\to \{v, \cap, e\} \\ e &\to \{v, \cap, v\} \\ e &\to \{e, \cap, e\} \\ &\cdots \\ v &\to \{x.id(t), =, y.id(t)\} \\ v &\to \{dist.L_2(x, y), >, dist.acc(x, y)\} \\ v &\to \{x.speed, >, s.speed\_limit\} \\ v &\to \{x.pos, L_2\_dist(x, y) < \theta, y.pos\} \\ &\cdots, \end{aligned} \tag{1}$$

where $e, \cap, v$ represent an expression (or operand), an operator (e.g., logical AND), and a value, respectively. An expression (e) can be replaced by a value, an operator, and an expression: ($\{v, o, e\}$). A value can be replaced by a physically meaningful computation on the input tracklets and/or geo-objects. For instance, a value (v) can be replaced by identity comparison ($\{x.id(t), =, y.id(t)\}$).

Since we generate a logic statement corresponding to the WHERE clause, the original problem, how to find a SQL statement to represent an activity, is cast to how to find a feasible set of rewriting rules and how to combine the rules.

This procedure is explained by a language generation (context-free language) using the type-2 grammar (or context-free grammar). A context-free grammar is defined by rules of the form $A \to \gamma$ with a nonterminal A and a string γ of terminals and nonterminals. This context-free languages are the theoretical basis for the phrase structure of most programming languages.

For simplicity, the right side of a rule containing value (v) can be expressed by a string. For example, the rule $v \to \{x.pos, L_2\_dist(x, y) < \theta, y.pos\}$ is implemented by $v \to$ "$((abs(T_1.latitude - G_1.latitude) + abs(T_1.longitude - G_1.longitude)) < 0.0001 \times 5)$", where T_1 is the first tracklet, G_1 is the first geo-object, and 0.0001×5 is a threshold controlling the distance between the tracklet and the geo-object. We use a set of different numbers for the thresholds (e.g., $0.0001 \times 1, 0.0001 \times 2, \cdots$) in our implementation so that our framework automatically finds reasonable thresholds for different types of activities.

4.2 Efficient random search with constraints

Our RDBMS (R), containing all extracted tracklets and geo-objects, returns an answer (a query result, (r_i)) given a SQL statement (q_i). We define a cost function ($f := f(r_i) = f(q_i|R)$) for a SQL statement, which allows us to examine all generated SQL statements. For efficient search, we propose to use MCMC (Markov Chain Monte Carlo) algorithm along with some techniques that reduce the search space.

4.2.1 MCMC (Markov Chain Monte Carlo)

The search algorithm is based on data-driven MCMC (Markov Chain Monte Carlo) method [Gilks 2005], which searches for an optimal SQL description to each input set of examples. The MCMC sampling generates samples by our searching heuristics and select a final one with the highest likelihood. The cost of each SQL statement is a function of the output results generated by the SQL and RDBMS.

4.2.2 Efficient search techniques

To reduce the size of the search space, we propose to use several techniques in our search framework.

Exclusive rules. We randomly generate a SQL statement from a set of rewriting rules. We begin from the starting expression (e) and replace it with one of all possible rules (e.g., $e \to (e, \cap, e) \to ((x == y), \cap, e) \cdots$. It is possible that a single SQL statement contains repeated expressions (e.g., $((x == y), \cap, (x == y)))$ or meaningless expressions (e.g., $((x == y), \cup, (x != y)))$. To avoid this redundancy or infeasible expressions, we define types of rules, such as *identity, geography, distance, turn*, group exclusive rules that cannot coexist in a single SQL, and sample only one of rules from the same type of rules for a SQL statement.

We also use an integer counter representing the possible number of occurrence in a single SQL statement. For instance, the identity comparison between two tracklets ($x.id(t) == y.id(t)$) should be used only once for the same pair of the tracklets whereas expansion of a binary relation $e \to (e, \cap, e)$ can happen multiple times (e.g., > 5).

SQL simplification and hashing. We define a specific set of rewriting rules to simplify a generated SQL statement. For instance, ($1 \cap 1$) is simplified by (1). We also use a hash table to avoid repeating evaluation of the identical logic representation.

4.3 Score function

We define a score function that maximizes the overlap between the result (R) and positive examples (Pos) while minimizing the overlap with the negative examples (Neg).

The overlap between (R) and (Pos) is defined by the ratio between the number of samples (tracklets) in the intersection of (R) and (Pos): $\frac{\sharp(Pos \cap R)}{\sharp(Pos)}$. Similarly, the overlapping (B) between (R) and (Neg) is defined as $(1 - \frac{\sharp(Neg \cap R)}{\sharp(Neg)})$. To control generalization, we define a function that returns null (ϕ) if the number of outputs is greater than a threshold:

$$\delta(c) = \begin{cases} 0 & \text{if } \sharp(R)/\sharp(S) > \theta \\ \lambda\sharp(R) & \text{else,} \end{cases} \tag{2}$$

where $\sharp(\cdot)$ and λ are the number of tracklets and a control parameter, respectively. Finally, the score is defined as $\delta(A + B)$.

5. EXPERIMENTAL RESULTS

We have implemented our framework and validated the approach on real data. In off-line processing step, we extracted tracklets with attributes and inserted them into a RDBMS. We also extracted geo-objects using *OpenStreetMap* from the corresponding area and inserted them into the database. In on-line testing step, given input positive/negative examples, our system runs the MCMC-based search module with an in-memory RDBMS, and provides a visualization result using Google Earth [Lisle 2006].

Data. We tested on a real dataset with simple motion patterns (e.g., *loop*, *3-point-turn*) and geospatial activities (e.g., *parking in a lot*, *leaving a parking lot*, and *in-n-out of a parking lot*). The dataset is a subset of tracking results extracted from the CLIF 2006 dataset [AFRL 2006]. The footprint of the area where we computed tracks is about 1 km^2, and its duration is about 8 minutes. Each track is on average 1 minute long. The total number of tracks estimated in the sequence of interest is more than 8000.

We manually defined parking lots and inserted into the database. Since a parking lot can have an arbitrary shape, we used a set of points to cover an entire parking area. Each point is stored with the parking lot ID in an off-line processing step.

We used 32 rewriting rules including "Is a parking lot" and "distance between tracklet and object". Each distance measure is represented with three levels of thresholds.

Finding valid sets. We tested whether our system returns valid sets given few positive/negative examples. We used a set of labeled data ("U-turn", "loop", "2-point-turn", "3-point-turn", "entry", and "exit" [Choi et al. 2012]) and inserted all data into the database. We selected $2 \sim 3$ typical positive examples and arbitrary selected (< 10) negative examples.

We first selected 2 typical positive examples for "loop" and selected 6 negative examples (2 "exit", 3 "entry", and

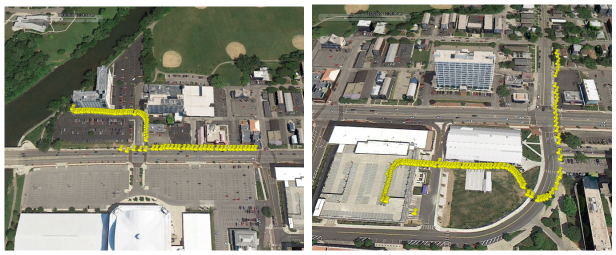

Figure 2: Examples of geospatial activities: "parking in a lot (id=16)" (left) and "leaving a parking lot (id=5987)" (right). The two tracks move toward different directions: right-to-left (left) and down-to-up (right).

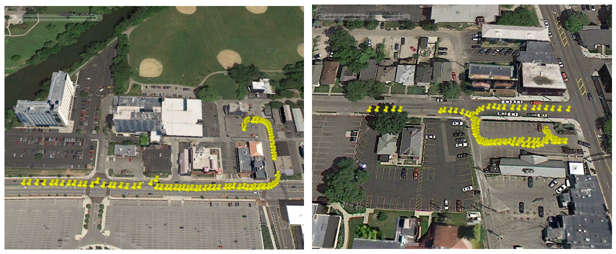

Figure 3: Identified results as "parking in a lot" using our proposed method: a correct result (left) and a wrong result (right). The car went to the lot and came our from the lot (right).

1 "3-point-turn"). In this experiment, we generated a large number of SQL statements, run each SQL, and evaluated result without using the MCMC framework. Our system generated $200K$ SQL statements and returned 58 $\sim$ 93 valid answers (0.029% $\sim$ 0.047%). Although many valid answers showed false alarms, we were able to reject these false alarms by using the result as new negative examples.

Geospatial activities. Here, we did not explicitly define 3 activities, using SQL, before conducting experiments. We first gathered a set of tracks that contain interactions between a vehicle and a parking lot as shown in Fig. 2. Assuming that "parking in a lot" is the target activity, we then selected few positive examples (id = {16, 34, 515}) and negative examples (id = {907, 5987, 42, 2116}) as input to our system.

Our system identified similar activities from entire database (id = {3846, 155, 20, ...}). Fig. 3 (left) (id = 3846) shows a correctly identified activity. However, the result included some false alarms (id = {62, 632}) as shown in Fig. 3 (right) (id = 632). These false accepted results "in-n-out of a parking lot" activities.

The reconstructed SQL WHERE clause is

$$\begin{aligned}(((T2.stop = 1 AND \\ ((abs(T2.latitude - G1.latitude)+ \\ abs(T2.longitude - G1.longitude)) < 0.0004 * 2)) AND \\ T2.frame - T1.frame > 30) AND \\ (((T1.track\_id = T2.track\_id) AND \\ G1.type = 100))),\end{aligned} \tag{3}$$

where $G1.type = 100$ means that the type of GIS object is "parking lot".

To show an interactive refinement, we added the wrong result (id = 632) as an additional negative example to the original set. After adding new negative examples, our systems returned improved results where all "in-n-out of a parking lot" activities were classified as negatives (id = {62, 632}). The updated SQL includes relations among three tracklets whereas the previous SQL concerns only relations between two tracklets.

We have also analyzed the descriptive power, the discriminant power, and the computational complexity of our approach. It seems that our approach can be applied to large scale real problems.

6. CONCLUSION

We presented an automatic system that learns symbolic representations of activities from examples in Wide Area Aerial Surveillance (WAAS). We validated our approach on real tracking data with motion patterns and complex geospatial activities. We will work on further validation with a large database and complex activities. We will also show the convergence property of our search framework.

Acknowledgment

This work was supported in part by grant DE-NA0001683 from the U.S. Department of Energy.

7. REFERENCES

[AFRL 2006] AFRL. 2006. Columbus Large Image Format (CLIF) 2006 Dataset Overview, https://www.sdms.afrl.af.mil. (2006).

[Agrawal et al. 1993] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. 1993. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, Vol. 22. ACM, 207–216.

[Choi et al. 2012] Jongmoo Choi, Yann Dumortier, Jan Prokaj, and Gérard Medioni. 2012. Activity recognition in wide aerial video surveillance using entity relationship models. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. ACM, 466–469.

[Chorowski and Zurada 2011] Jan Chorowski and Jacek M Zurada. 2011. Extracting rules from neural networks as decision diagrams. *Neural Networks, IEEE Transactions on* 22, 12 (2011), 2435–2446.

[Giannatos et al. 2012] Spyridon Giannatos, Yun-Gyung Cheong, Mark J Nelson, and Georgios N Yannakakis. 2012. Generating narrative action schemas for suspense. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference.*

[Gilks 2005] Walter R Gilks. 2005. *Markov chain monte carlo.* Wiley Online Library.

[Haklay and Weber 2008] Mordechai Haklay and Patrick Weber. 2008. Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE* 7, 4 (2008), 12–18.

[Lisle 2006] Richard J Lisle. 2006. Google Earth: a new geological resource. *Geology today* 22, 1 (2006), 29–32.

[Prokaj et al. 2011] J. Prokaj, M. Duchaineau, and G. Medioni. 2011. Inferring tracklets for multi-object tracking. In *CVPRW*. 37–44.