Identifying Inhabitants of an Intelligent Environment Using a Graph-Based Data Mining System

Ritesh Mehta, Diane J. Cook and Lawrence B. Holder

Department of Computer Science and Engineering
University of Texas at Arlington
Box 19015, Arlington, Texas 76019-0015
{mehta, cook}@cse.uta.edu

Abstract

The goal of the MavHome smart home project is to build an intelligent home environment that is aware of its inhabitants and their activities. Such a home is designed to provide maximum comfort to inhabitants at minimum cost. This can be done by learning the activities of the inhabitants and to automate those activities. For this it is necessary to identify among multiple inhabitants who is currently present in the home. Subdue is a graph-based data mining algorithm that discovers patterns in structural data. By representing the activity patterns for each inhabitant as graphs, Subdue can be used for inhabitant identification. We introduce a multiple-class learning version of Subdue and show some preliminary results on synthetic smart home activity data for multiple inhabitants.

Keywords: Data Mining, Machine Learning, Intelligent Environments, Graph Representation, Minimum Description Length

Introduction

Recent advances in machine learning research and application motivate us to construct a smart home that provides maximum comfort to its inhabitants. One use of today's technology is to automate day-to-day activities of individuals with minimum interaction and minimal operating costs. This is the goal of the MavHome smart home project (Das et al. 2002). MavHome learns patterns of inhabitant activity and automates selected activities through control of home devices.

Activities performed by inhabitants, such as switching on a floor lamp in the living room, can be logged. Over time, this collected information becomes a large data set that contains extensive knowledge useful for learning. Mining of data has become an important research technique due to its ability to extract knowledge from large databases. Mining can, in the context of our work, be used for discovering patterns representing inhabitant's activities from smart home data.

Smart home data is structural in nature, or is composed of data and relationships between the data. The data points in this application consist of individual inhabitant activities and the data is related spatially and temporally. Therefore, we need a data mining technique that can represent this structural information. Individual activity data points, and relationships between these data points, can be represented as a graph. Subdue is a graph-based data mining algorithm (Cook and Holder 2000). The distinct advantage of Subdue as compared to other data mining techniques is its ability to mine structural information. Individual data points can be mapped to vertices and relationships between the data can be mapped to edges in a graph.

In order to provide maximum comfort, MavHome needs to automate selected inhabitant activities. By learning patterns of typical inhabitant activities, MavHome can automate some of the interactions with the house. This task is complicated if there are multiple inhabitants in a house. Therefore, the initial task is to perform inhabitant identification and then perform activity prediction.

Subdue has many features such as a supervised concept learner which can be used for inhabitant identification. The Subdue supervised concept learner, SubdueCL, supports binary classification. We modify this system to handle classification of inhabitants among an arbitrary number of possibilities, within the MavHome smart home project.

This paper is organized as follows. We first describe the MavHome project, followed by background on Subdue and SubdueCL. We then introduce the Multiple-Class Learning using Subdue (SubdueCLM) algorithm and validate our approach with empirical evidence. Finally we conclude with observations and directions for future research.

The MavHome Smart Home Environment

The MavHome (Managing an Intelligent Versatile Home) smart home project is a multi-disciplinary research project at the University of Texas at Arlington focused on the creation of an intelligent home environment. The goal of the home is to create an environment that is aware of its inhabitants and activities. One of the tasks of MavHome is to identify the inhabitants, learn their activities and automate them, so that it can maximize comfort to the inhabitants and minimize the cost of operations and maintenance. In order to meet these goals, the home should continuously learn and adapt to the changing activities of the inhabitants. Our approach is to view the smart home as an intelligent agent that perceives its environment through the use of sensors, and can act upon the environment through the use of actuators. In short, the home should be able to correctly distinguish between multiple inhabitants, predict their activities, and select some of the activities for automation. This information can be used by the home to pass necessary messages to appropriate devices in order to automate activities which otherwise would be performed by the inhabitants manually.

MavHome operations can be characterized by the following scenario (Das et al. 2002). At 6:45am, MavHome turns up the heat because it has learned that the home needs 15 minutes to warm to optimal temperature for waking. The alarm goes off at 7:00, which signals the bedroom light to go on as well as the coffee maker in the kitchen. One of MavHome's inhabitants, Bob, steps into the bathroom and turns on the light. MavHome records this interaction, displays the morning news on the bathroom video screen, and turns on the shower. While Bob is shaving MavHome senses that Bob is two pounds over his ideal weight and adjusts Bob's suggested menu. When Bob finishes grooming, the bathroom light turns off while the kitchen light and menu/schedule display turns on. During breakfast, Bob notices that the floor is dirty and requests the janitor robot to clean the house. When Bob leaves for work, MavHome secures the home, and starts the lawn sprinklers despite knowing the 70% predicted chance of rain.

This scenario requires a number of tasks to be performed, including data collection, data mining, and activity prediction, as well as information passing between multiple agents. The task of activity prediction is further complicated if there is more than one inhabitant in a house. Different inhabitants follow different activity patterns and activities for some inhabitants may overlap with others. For example, after Bob steps out of the bathroom, another MavHome inhabitant, Jack, steps into the bathroom and turns on the light. This activity of Jack overlaps with Bob, so the home should be able to distinguish between the Bob and Jack activities.

As the scenario suggests, MavHome needs to first identify who is currently in the house and then predict the inhabitants' activities. In the case of multiple inhabitants present at the same time, MavHome needs to distinguish between the inhabitants' activities. We hypothesize that this identification of inhabitants can be performed given a history of device interactions between inhabitants and household devices. Due to the structural nature of the data, Subdue is particularly well suited to the task.

Subdue

Subdue (Cook and Holder 2000) is a graph-based data mining tool that can discover patterns and learn concepts from structured data. It is a general tool that can be applied to any domain that can be represented as a graph. Subdue expects input to be a labeled graph. Objects in the data are mapped to vertices and relationships between objects are mapped to directed or undirected edges.

The main algorithm for discovery is a variant of a beam search. The goal of the search algorithm is to find a subgraph that compresses the input graph the best. These subgraphs, or substructures, are evaluated according to Minimum Description Length principle, originally developed by Rissanen (Rissanen 1989). This compression is calculated as follows:

$$Compression = \frac{DL(S) + DL(G \mid S)}{DL(G)}$$

where DL(G) is the description length of the input graph, DL(S) is the description length of the subgraph and DL(G|S) is the description length of the input graph compressed by the subgraph. The search algorithm tries to maximize the value of the subgraph, which is simply the inverse of the compression.

The initial state of the search is the set of subgraphs consisting of every part of uniquely-labeled vertices, where each subgraph represents a uniquely-labeled vertex. The search begins by extending a subgraph in all possible ways by a single edge and a vertex, or by a single edge if both vertices are already in the subgraph. The algorithm searches for the best substructure until all possible substructures have been considered or the total amount of computation exceeds a given limit.

Once the search terminates and returns a list of best subgraphs ordered by compression value, the graph can be actually compressed using the best subgraph. The compression procedure replaces all instances of the subgraph with a pointer to the newly discovered substructure. The discovered substructures allow abstraction over detail in the original data. The resulting graph can be input to another iteration of Subdue, which further compresses the graph. This process of substructure discovery and compression can be performed on the graph until it cannot be further compressed (i.e., the graph is compressed into a single vertex). After several iterations, Subdue builds a hierarchical description of the structural data where substructures discovered later in the process are defined in terms of substructures discovered during earlier iterations.

Subdue also has a feature by which predefined substructures can be provided to Subdue. In this predefined mode, Subdue will try to find and expand predefined substructure instances. Subdue reports to the user whether instances of the predefined substructure occur in the input graph.

Figure 1 shows a simple example of Subdue's operation. Subdue finds four instances of the triangle-on-square substructure in the geometric figure. The graph representation used to describe the substructure, as well as the input graph, is shown.

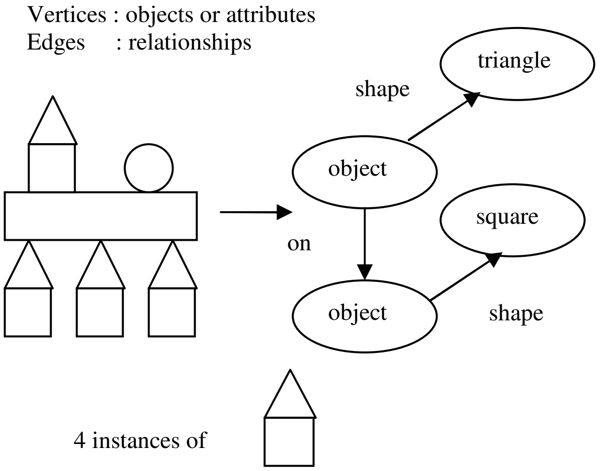

Figure 1: Subdue Example

The Subdue Concept Learner

Concept learning is a process that consists of the induction of a function (concept) from training examples. Training examples are labeled as belonging to class x if they contain the substructures discovered for concept x. The training examples are used to guide the search for the target function. After the learning process, this function must be capable of correctly classifying a new example, one that was not included in the training examples. Then, if the training examples are an accurate representation of their domain, it is possible to learn a function that will be accurate when tested with new examples.

Subdue can act as a supervised concept learner (Cook and Holder 2000). The Subdue concept learner (SubdueCL) accepts positive and negative examples in graph format. Substructures that occur often in the positive graph but not often in the negative graph are likely to be the target function. Therefore SubdueCL discovers substructures that compress the positive graph more than the negative graph.

The compression value for a substructure S is calculated as follows:

$$Value(G_p, G_n, S) = DL(G_p, S) + DL(S) + DL(G_n) - DL(G_n, S)$$

where DL(G,S) is the description length, according to the MDL encoding, of a graph G after being compressed using substructure S and DL(G) is description length of a graph G. This value represents information needed to represent the positive graph Gp using the substructure S plus the information needed to represent the portion of the negative graph Gn that was compressed using substructure S. SubdueCL will iterate until it finds a substructure that compresses the positive graph more than the negative graph.

One of the limitations of this compression-based concept learner is that it only looks for substructures which compress the entire positive graph more than the entire negative graph. Therefore, it is biased to look for a substructure that offers more compression as compared to a substructure that covers a greater number of positive examples.

The Subdue set-covering approach (Gonzalez, Holder and Cook 2001) to concept learning looks for substructures that cover the greatest possible number of positive examples while not covering negative examples. The evaluation of substructure S thus becomes

$$Value = \frac{\#PosEgsCovered + \#NegEgsNotCovered}{\#PosEgs + \#NegEgs}$$

where #PosEgsCovered is the number of positive examples covered by the substructure and #NegEgsNotCovered is the number of negative examples not covered by the substructure. #PosEgs is the total number of positive examples and #NegEgs is the total number of negative examples.

Multiple-Class Learning Using Subdue

To date, SubdueCL can perform only binary classification. However, in the smart home task a classification needs to be made among multiple possible inhabitants. We have extended Subdue to learn concepts with multi-valued target attributes. This algorithm, SubdueCLM, expects an input file that contains the number of classes that are possible as well as labeled training data for each class.

SubdueCLM uses SubdueCL to perform classification separately for each value of the target attribute (in this case, the inhabitant label). The steps are

1. Read and create a graph for each class. While reading, SubdueCLM converts the logged smart home activity data to a graph.
2. Iterate for the number of classes specified. During each iteration, SubdueCLM views the data for one class value (one inhabitant) as a positive graph and the collection of the other graphs as a negative graph. If there are more than two classes, the graphs representing all classifications except the current target value are merged into one negative graph. Substructures are then discovered for these positive and negative graphs using SubdueCL.

3. Store the concept found during each iteration for classification of new data points.

This algorithm will find concepts that represent a target function for each class. The target function can now be used to perform classification among previously-unseen examples. In order to perform classification, the new inhabitant activity data is converted to a graph. Using the ability to search for a predefined substructure provided by Subdue, SubdueCLM can search for the substructures representing each class value in the new data. If one of the learned structures is found in the input graph, the corresponding class value (in this case, inhabitant label) is reported. Because a smart home may have multiple inhabitants at home at the same time, all inhabitants whose patterns are noticed in the data are reported as currently being present.

Identification of Multiple Inhabitants in MavHome Using Subdue.

Subdue can be used for identification of inhabitants in MavHome. A smart home can have many inhabitants. Therefore, in order to automate activities for each individual, the home must identify them. SubdueCLM can be used for identification of a home's inhabitants from observed interactions with the home. In order to use SubdueCLM we must convert MavHome data to a graph representation.

An event in a smart home consists of interaction with a device, such as turning a lamp on or off, playing music, and starting the water sprinklers. Each event can be considered as a device whose state is being changed at a particular time. The event device and event time can be mapped to nodes with an edge labeled with the state change.

In the smart home graph, consecutive events are connected with an edge that is labeled with the discretized time difference between the two events. For example, if the time difference is less than 5 minutes than the edge is labeled as ImmediatelyFollows, if the time difference is more than 5 minutes but less than 15 minutes the edge is labeled as SoonFollows and so on. The amount of discretization can be varied according to the application.

A simple example of MavHome data is as follows. The data follows the format collected by software packages such as HomeSeer.

(1) 7/7/2001 9:52:07 AM~!~X10 Received~!~A6 (?) A On

(2) 7/7/2001 9:55:07 AM~!~X10 Received~!~A6 (?) A Off

Entries (1) And (2) represents two events. The first entry indicates that lamp A6 was switched ON at 9:52 am on 7/7/2001 and was switched OFF at 9:55am on the same day. The corresponding graph representation is shown in Figure 2.

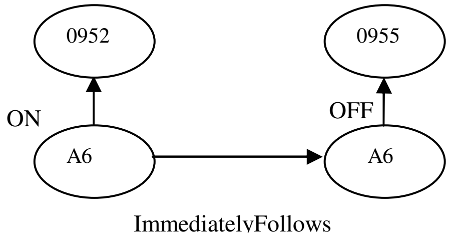

Figure 2: Graph representation of MavHome data

The activity log for each inhabitant collected over a number of days can be used to train SubdueCLM. SubdueCLM will find concepts describing each inhabitant's activity pattern. This concept can now be used to classify new activity logs which were not used to train the system. In this way, inhabitants can be identified according to the activity they perform with the house.

Experimental Results

To illustrate the classification of inhabitants using SubdueCLM, we apply the SubdueCLM algorithm on synthetic data for three inhabitants, which indicates activities followed by each inhabitant. The activity patterns for two of the inhabitants, Ritesh and Sira, overlaps in time. As a result, a pattern for one individual may be interrupted by activities for another individual. Activity patterns for the third inhabitant, Karthik, do not overlap the others.

The primary activity pattern followed by Ritesh is to enter the environment, play some music, turn ON lights on way to his desk and at his desk. After some time, Ritesh goes to the lounge and watches TV. Before leaving, Ritesh turns off the TV, the lounge light and the light at his desk.

The activity pattern for Sira is similar to that for Ritesh. Sira enters the environment 15 to 20 minutes after Ritesh enters, plays some music and turns ON lights at his desk. Other activities which Sira performs are to close the blinds, go to the kitchen and select some food to eat from the refrigerator. Before leaving, Sira switches off the lights at his desk and in the pathway. Ritesh and Sira are together in the lab for 20 to 25 minutes.

Karthik's activity pattern does not overlap with other two. After entering the smart environment, Karthik plays some music, turns on lights on the way to his desk and opens the blinds. Upon entering the kitchen, Karthik turns on the kitchen light, opens the refrigerator, and turns off the light before leaving the kitchen. Before leaving environment, Karthik watches TV for a while and finally closes the blinds and switches off the light in the pathway.

The graph representation used for our synthetic data is as explained in previous section. We use separate labels for consecutive events occurring within 5 minutes, 15 minutes, 60 minutes, and 120 minutes of each other

SubdueCLM is tested using the compression evaluation heuristic as well as using the set-covering evaluation heuristic. Experiments used ten days of activity log for each inhabitant, where seven days worth of data is used for training and the remaining three days is used for testing. Subdue version 5.0.3 is used for our experiment.

The predictive accuracy for SubdueCLM approach is 100% with both evaluation heuristics if log for each inhabitant is tested individually. We also have tested SubdueCLM by interleaving activity logs for multiple inhabitants. The interleaving of data is performed in two ways. In the first case, an activity log is interleaved according to the time each event occurs in a day. In this case, the predictive accuracy is 0% for the Ritesh and Sira whereas it is 100% for Karthik using the compression heuristic. For the set-covering approach the predictive accuracy is 0% for Ritesh, 66.67% for Sira and 100% for Karthik. In the second case, the activity log is interleaved according to inhabitant. This means that the log file for three days of activity for Ritesh is followed by three days of activity for Sira, and so on. The predictive accuracy in this case was found to be 100% for each inhabitant for both approaches.

If the relative order of the events in the activity pattern for inhabitants is changed then SubdueCLM does not perform well. Results for MavHome environment show that SubdueCLM can classify inhabitants correctly if the inhabitant follows a sequential activity pattern.

Conclusions

In this paper we introduced an algorithm, SubdueCLM, for classification of multi-valued concepts using the graph-based data mining system Subdue. We demonstrate the application of SubdueCLM to data from the MavHome smart home project. The experiments performed using MavHome data to classify inhabitants based on activity log show that SubdueCLM can successfully identify inhabitants from observing inhabitant activity.

There are some enhancements that we would like to make to SubdueCLM to increase its usefulness for the MavHome project. We need to refine SubdueCLM to perform classification when the smart home the activity log contains partially-ordered events and when activities for multiple inhabitants overlap. We would like to perform an exhaustive testing for a different number of inhabitants with different activity patterns and different amounts of overlapping between inhabitant activity patterns. We are currently gathering real activity data that will be used for in-depth testing of the algorithm. We would also like to apply SubdueCLM to other domains which can benefit from a structural approach to multiple-class concept learning.

Acknowledgements

This work was supported by National Science Foundation grant IIS-0121297.

References

Cook, D. J., and Holder, L. B. 2000. Graph-Based Data Mining. *IEEE Intelligent Systems*, 15(2):32-41.

Gonzalez, J., Holder, L. B., and Cook, D. J. 2001. Graph-Based Concept Learning. In *Proceedings of the Florida Artificial Intelligence Research Symposium,*377-381.

Jonyer, Holder, L. B., and Cook, D. J. 2000. Graph-Based Hierarchical Conceptual Clustering. In *Proceedings of the Florida AI Research Symposium*,91—95.

Das, S. K., Cook, D. J., Bhattacharya, A., Heierman, E. O. III, and Lin, T.-Y. 2002. The Role of Prediction Algorithms in the MavHome Smart Home Architecture. *IEEE Personal Communications Special Issue on Smart Homes*, Vol 9, No. 6: 77-84.

Gonzalez, J., Jonyer, I., Holder, L. B., and Cook, D. J. 2000. Efficient Mining of Graph-Based Data. *Proceedings of the AAAI Workshop on Learning Statistical Models from Relational Data*, 21-28.

Jonyer, Cook, D. J., and Holder, L. B. 2001. Discovery and Evaluation of Graph-Based Hierarchical Conceptual Clusters. *Journal of Machine Learning Research*, 2:19-43.

Rissanen, J. 1989. Stochastic Complexity in Statistical Inquiry. World Scientific Publishing Company.