Graph-based Learning for Smart Environments

S. Anderson, S. Long, L. Holder

Electrical Engineering and Computer Science, Washington State University, Pullman, WA

Project Overview

About CASAS

CASAS is a multidisciplinary research project at WSU, the focused on creating an automated living environment. The chief goal is the application of Smart Environment technology in helping elderly inhabitants continue to live independently at home. Aspects of the project range from the purely electronic (sensor systems, etc.) to medical and ergonomic research. Besides its implications for machine learning in general, our research is significant to living assistance, as it could be used track the activities of Smart Home occupants, recognize when they have difficulties with day to day tasks, and allow other systems of the house to respond appropriately.

Problem: Classifying Activities from Motion Sensor Data

The challenge of activity recognition in Smart Environments represents a major challenge in machine learning.

- Unpredictable behavior
- Complex tasks
- Multiple inhabitants
- Multitasking
- Privacy concerns limit tracking systems

In the CASAS Smart Home project, occupants of an apartment are tracked using motion sensors. The objective is to create an algorithm that can identify when the occupants perform one of ten activities.

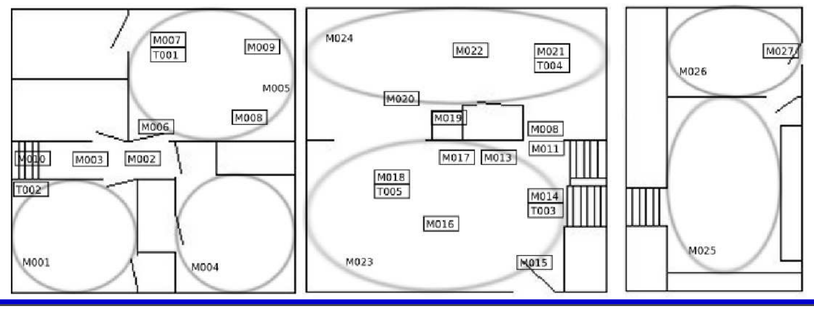
Graph Based Learning

- Machine learning for activity classification
- Graphs emphasize relationships in data
- Use graphs to obtain information for machine learning algorithm

Our goal: maximize accuracy with a graph-based algorithm

Activities (number):

- Bed to Bathroom (30)
- Breakfast (48)
- Dinner (42)
- Laundry (10)
- Leave Home (69)
- Lunch (37)
- Night Wandering (67)
- Sleep (102)
- Take Medicine (44)
- Work in Office (46)



Floor plan of the test apartment.
M0* - Motion Sensor No Box - Area Motion Sensor T0* - Temperature Sensor

The Algorithm

Regardless of the additional methods used to increase accuracy, the machine learning algorithm has three basic steps. First, we create graphs of the data, where edges represent motion between sensors, and analyze the graphs to find significant subgraphs. These subgraphs are then used as feature vectors for an SVM model. Finally, the SVM is trained and testing using cross-validation.

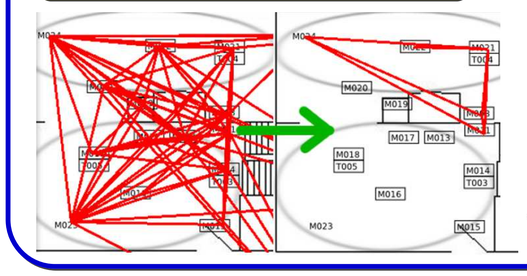
Frequent Subgraph Mining

We use Gaston [2], to find common subgraphs across the activity. The most frequent subgraphs are collected and added to the feature vector.

OR

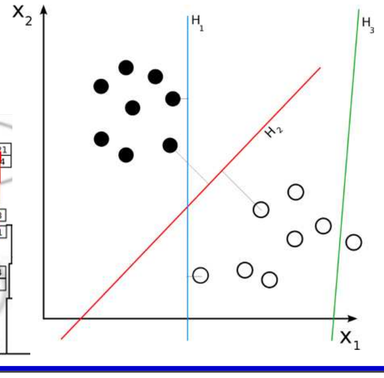
Minimum Description Length

Alternative to Frequent Subgraph mining: using the MDL criterion, with Subdue [1]. The "best" subgraphs are those that make up a large portion of the graphs. This strikes a balance between large and common subgraphs.



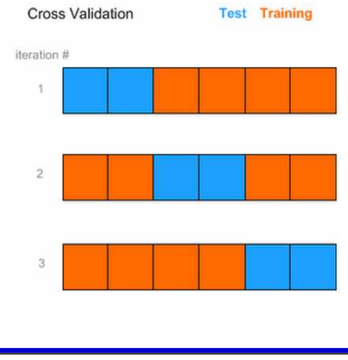
Support Vector Machines

The algorithm uses Support Vector Machines with a Radial Bias Function kernel, a leading method for general classification problems [3]. The key to creating an effective SVM is appropriate choice of features by which to identify examples. Our attempts to improve the accuracy of the algorithm focus on testing various ways to obtain features, using graph mining.



Cross Validation

In a real machine learning application, we would have a training set of labeled data, and new, unlabeled data to classify. To simulate this, and estimate the accuracy of the algorithm, we use 3-cross validation. The data are split into three sets, and the algorithm is run three times with one set as the test data and the others as the training data. Resulting accuracies are averaged.



Algorithm Improvements

Negative Examples

The Subdue program's Minimum Description Length graph mining algorithm supports the inclusion of negative examples - examples of graphs that do not match the current activity. The software attempts to find subgraphs that compress the positive examples, but not the negative. We hypothesized that using this technique with a sampling of negative examples would improve accuracy, but the results were not significantly improved.

Removing Unlikely Connections

It is possible for data that do not correspond to the current activity to throw off the learning algorithm. There are many cases in the test data in which a sensor is activated that could not have been feasibly reached from the previously active sensor, possibly due to the presence of multiple individuals in the apartment, or from environmental motion. One effort toward improving accuracy was to eliminate connections between sensors that should not reasonably be connected.

Time Information

Moving beyond a purely graph-based approach, including the start and end time of the activities in their feature vectors provides an extra piece of information that would help identify them, for example distinguishing breakfast, lunch, and dinner.
We attempted an alternative way of taking time into account by using Dynamic Graphs - tracking the changes in the graph over time, and searching periodically recurring subgraphs. This proved not to be viable for this particular application.

Results

	Non-Graph SVM	Frequent Subgraph SVM	MDL SVM
Random	20.61 %	20.61 %	20.61 %
Base Accuracy	68.69 %	58.79 %	58.79 %
Negative Examples	n/a	n/a	55.76 %
Connection Removal	n/a	64.65 %	64.85 %
Append Time	69.90 %	65.05 %	65.05 %

Large scale use of negative examples reduced the accuracy of MDL search slightly, and at small scale had no noticeable effect. Removing just a few unlikely connections, however, caused a significant increase in both graph-based algorithms. Adding the start and end times to the feature vector increased all methods, but especially the non-graph SVM.

Periodic Graph Random	Best Achieved
7.6 %	8.2 %

The dynamic graph method we attempted barely exceeded the accuracy of randomly guessing.

Conclusion

In spite of improvements to the graph-based algorithms, they are still less accurate than the non-graph method. These results suggest that:

- Complex methods of creating feature vectors (e.g., dynamic graphs) give too specific results to work in machine learning
- Successful connection removal could be used for pre-processing
- This would risk over-fitting
- Non graph based information such as time can be successfully incorporated to a graph-based algorithm

Future Work:

- Test pre-processing with connection removal (requires altering test system)
- Investigating pre-processing with negative examples on small scale could still work
- Attempt recognition of activity boundaries in addition to activities themselves

References

[1] Diane J. Cook and Lawrence B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15:32–41, March 2000.

[2] Siegfried Nijssen and Joost N. Kok. The Gaston tool for frequent subgraph mining. *Electron. Notes Theor. Comput. Sci.*, 127:77–87, March 2005.

[3] B. Scholkopf, Kah-Kay Sung, C.J.C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *Signal Processing, IEEE Transactions on*, 45(11):2758 –2765, nov 1997.

This work was supported by the National Science Foundations REU program under grant IIS-0647705.