Real-Time Occupancy Detection using Decision Trees with Multiple Sensor Types

Ebenezer Hailemariam, Rhys Goldstein, Ramtin Attar, Azam Khan

Autodesk Research
210 King St. East, Toronto, Ontario, Canada, M5A 1J7
{firstname.lastname}@autodesk.com

Keywords: Occupancy detection, sensor data, classification, decision tree, building performance

Abstract

The ability to accurately determine localized building occupancy in real time enables several compelling applications, including intelligent control of building systems to minimize energy use and real-time building visualization. Having equipped an office workspace with a heterogeneous sensor array, our goal was to use the sensors in tandem to produce a real-time occupancy detector. We used Decision Trees to perform the classification and to explore the relationship between different types of sensors, features derived from sensor data, and occupancy.

We found that the individual feature which best distinguished presence from absence was the root mean square error of a passive infrared motion sensor, calculated over a two-minute period. When used with a simple threshold, this individual feature detected occupancy with 97.9% accuracy. Combining multiple motion sensor features with a decision tree, the accuracy improved to 98.4%. Counterintuitively, the addition of other types of sensors, such as sound, CO_2, and power use, worsened the classification results. The implication is that, while Decision Trees may improve occupancy detection systems based on motion sensors alone, one risks overfitting if multiple types of sensors are combined.

1 INTRODUCTION

In addressing building performance, we face a complicated balance between occupant comfort and energy consumption, as comfort and energy savings are often inversely related [Yilmaz, 2007]. If we are to achieve this balance, intelligent building systems are needed that are aware of not only their occupants' environment, but also the number of occupants within that environment. In most office buildings, the office cubicle is what demarcates an occupant's personal workspace. It is within these semi-open spaces that we attempt to detect occupancy.

The ability to accurately determine localized building occupancy in real time enables several compelling applications; examples include the intelligent control of building systems, fine-grained space utilization data collection, real-time building visualization [Hailemariam et al., 2010], and security systems. An occupant detector dedicated to a specific cubicle could be used to inform a Personal Environmental Module, a device that controls the environment around a single office worker [Mendler et al., 2006]. With an occupant detector at nearly every workspace, a central building control system could minimize energy wasted in the heating, air conditioning, and lighting of unoccupied spaces.

Currently, most commercial systems which perform occupant detection solely utilize passive infrared (PIR) motion detectors. These systems apply relatively simple analysis to the sensor signal to infer occupancy from the degree of apparent motion. One problem with these systems is that PIR sensors fail to detect subjects which remain relatively still. Furthermore, distant passersby and wafts of warm or cold air are interpreted as motion leading to false positives. Traditional approaches also rely on camera-based methods [Ramoser et al., 2003]. However, these methods require complex video processing and introduce privacy concerns.

In our approach we embed a number of low-cost sensors of different types into the cubicle furniture. Using these sensors we measure several attributes of the occupant's environment: light, sound, CO_2 level, power use, and motion. Through Decision Tree analysis we develop a method to deduce the occupancy of the workspace at any given time.

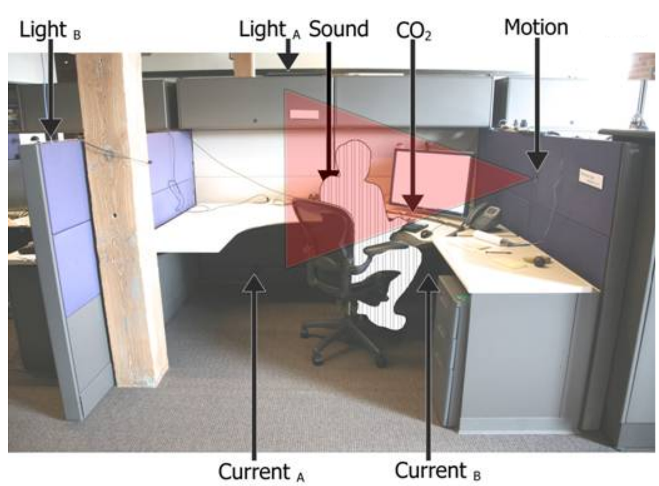

Figure 1. Physical configuration of sensors installed in a cubicle to help detect occupancy. The red cone denotes the sensing region of the motion sensor.

Our approach is not without precedent. The authors of [Tarzia et al., 2009] use sonar along with a simple threshold-based classifier to develop a computer which can detect the physical presence of a user. Similarly, the authors in [Dodier et al., 2006] deployed multiple passive infrared sensors into enclosed offices and used Bayesian probability theory to develop an occupancy classifier.

In [Lam et al., 2009a] and [Lam et al., 2009b], the authors deployed a wider array of sensor types (various gas sensors, sound pressure level, illuminance, PIR, temperature and humidity) into an open office environment to count the number of occupants. The classification methods used by Lam et al. were *Support Vector Machines (SVM)*, *Neural Networks*, and *Hidden Markov Models (HMM)*. Features were selected prior to classification based on *information gain*, a statistical quantity commonly used to measure the degree to which uncertainty is reduced by exploiting additional data.

We also treat occupancy detection as a classification problem, but use *Decision Trees* instead of the methods above. The simplicity of the Decision Tree method allows us to explore relationships in the sensor data. Our approach builds on previous work, focusing on the following:

- Occupant detection, as opposed to occupant counting.
- Occupant detection within a semi-open space (an office cubicle).
- The use of multiple low-cost sensor types to improve the detection quality over a single sensor alone.
- The use of Decision Trees to automatically select the features with the highest information gain and exploit those features to construct an occupancy classifier.
- The use of Decision Trees to explore the relationship between different types of sensors, features derived from sensor data, and occupancy.
- The use of various performance metrics to rank sensor types and features.

It is important to note that different applications of an occupant detector require different levels of accuracy. Whereas an HVAC system may be able to maintain appropriate temperature levels given relatively coarse real-time occupancy information, a security system or a power usage visualization tool may need to detect even short periods of presence and absence. Our interest lies in comparing the relative accuracies of various occupancy detection options, and so we do not seek any particular level of accuracy.

2 APPARATUS

We equipped a cubicle in our office with sensors to measure various attributes of the occupant's environment. Specifically, the sensors we chose measure those attributes which we believed could be directly correlated with occupancy and at the same time provide information about the environment which may be useful for other applications. The selected sensors measured carbon dioxide (CO_2), electrical current (current), illuminance (light), PIR (motion), and sound pressure level (sound). Table 1 describes the make, model and quantity of each sensor we deployed in our test bed. The physical placement of each of these sensors is illustrated in Figure 1.

Type	Make	Model	Qty.
CO_2	Sensair AB	k30	1
Current	Elkor Technologies Inc.	iSnail-VC-25	2
Light	Phidgets Inc.	1127	2
Motion	Phidgets Inc.	1111	1
Sound	Phidgets Inc.	1133	1
Total			7

Table 1. Sensors used in cubicle during the study.

Current sensor A ($Current_A$) measures the current drawn by a computer monitor with a 15 minute power saving timer enabled. Current Sensor B ($Current_B$) measures the current drawn by a desktop computer. Light sensor A ($Light_A$) and light sensor B ($Light_B$) are positioned to capture the conditions of the shared overhead lighting system. The motion sensor is placed such that its sensing region cuts the enclosed space of the cubicle thereby eliminating unwanted interference from passersby. The CO_2 sensor is located atop the occupant's desk between their keyboard and monitor so as to optimally capture exhaled CO_2. Lastly, the sound sensor is positioned so as to capture the voice of the occupant and their visitors from multiple directions while minimizing occlusion.

3 DATA COLLECTION

For our experiment we collected data for a single cubicle in our office over a contiguous seven day period, including weekends and business off hours. Ground truth occupancy data was obtained by recording the occupant using a conventional camera. A human operator would periodically review the video and produce a schedule, accurate to the minute, of when the occupant was present in their cubicle.

Sensor data was represented by sequences of time-value pairs. For two consecutive pairs $< t_n, v_n >, < t_{n+1}, v_{n+1} >$, value v_n spans the time interval $[t_n, t_{n+1})$. A new time-value pair would only be output if the value changed by a fixed sensor-dependent amount. Thus the sampling rate varied on a per sample basis depending on the variability of the sensor data. Noisy data, such as that produced by motion or sound sensors, would typically be acquired at the maximum sample rate of 2 Hz. The CO_2 sensor, light sensors, and computer current meter were captured at moderate sample rates. The relatively stable computer monitor current meter output relatively few time-value pairs. Over the testing period we collected roughly 41 million data points, as indicated in Table 2.

Type	Number of Data Points
CO_2	2, 233, 542
Current_A	132, 358
Current_B	3, 980, 179
Light_A	1, 137, 060
Light_B	1, 361, 186
Motion	14, 066, 006
Sound	18, 889, 586
Total	41, 799, 917

Table 2. Number of raw sensor data samples collected over seven consecutive days.

4 OCCUPANCY DETECTION FEATURES

Occupancy detection can be formulated as a classification problem in which a set of features must be automatically associated with one of several classes. In this context, a *feature* is a function of sensor data over a period of time, and a *class* is a representation of occupancy at a single point in time.

The simplest classifiers for occupancy detection use only a single feature and two classes. In [Padmanabh et al., 2009], for example, the state of a conference room is classified based on the number of times a microphone value exceeds a threshold in a 5-minute time interval (the feature). If the threshold is exceeded more than twice, a meeting is assumed to be in progress (one class); otherwise, the conference room is assumed to be in the "no meeting" state (the other class).

Our approach is similar to [Lam et al., 2009a] in that it exploits numerous features derived from multiple sensor types. However, because the classifiers in [Lam et al., 2009a] aim to count the number of occupants in a space, separate classes are used for vacancy, 1 person, 2 people, 3 people, etc. We use only two classes: one which represents the absence of any occupants and one which represents the presence of at least one occupant.

Expressions of the following form denote the value of a feature at time t:

$$\langle \text{type of feature} \rangle_{2^j}(\langle \text{type of sensor} \rangle)[t]$$

The integer j indicates that the feature value is calculated over a time period starting at time $(t - 2^j{\cdot}\Delta t)$ and ending at time t, where Δt is an arbitrary duration. With this notation and a Δt value of 1 minute, for example, $AVG_{2^5}(CO_2)[11:58]$ represents the average value of a carbon dioxide sensor from 11:26 to 11:58. Note that the time period over which a feature value is calculated precedes the time label t. Our focus on real-time applications motivates the use of only past information to detect occupancy.

Certain types of features provide more information when applied to certain types of sensors. That said, our approach is to apply every type of feature to every type of sensor. The task of identifying the most informative sensor-type/feature-type combinations is left to the classification method, as described later.

The equations that follow define our feature types, with X representing any sensor and $X(t)$ denoting the sensor reading at time t. The first feature we calculate is the average value over a duration of Δt.

$$AVG_{2^0}(X)[t] = \frac{1}{\Delta t}\cdot\int_{t-\Delta t}^{t} X(t)dt$$

As a sensor's raw data can often be interpreted as a step function, the integral above can generally be evaluated as a sum of duration-weighted sensor readings. Similarly, the integral in the $(j=0)$ root mean square (RMS) error feature below can in most cases be treated as a weighted sum of squares.

$$RMS_{2^0}(X)[t] = \sqrt{\frac{1}{\Delta t}\cdot\int_{t-\Delta t}^{t} (X(t))^2\,dt - (AVG_{2^0}(X)[t])^2}$$

The rationale for using time interval widths based on powers of 2 is that the $(j+1)^{\text{th}}$ set of features can be obtained efficiently from the j^{th} set. We define an upper limit j_{max} to constrain the number of feature values. Below are the *AVG* and *RMS* feature calculations for time intervals of duration $2{\cdot}\Delta t$, $4{\cdot}\Delta t$, $8{\cdot}\Delta t$, ..., $2^{j_{max}}{\cdot}\Delta t$.

$$AVG_{2^{j+1}}(X)[t] = \frac{AVG_{2^j}(X)[t-2^j{\cdot}\Delta t] + AVG_{2^j}(X)[t]}{2}$$

$$RMS_{2^{j+1}}(X)[t] = \sqrt{\frac{MS+MSX}{2} - (AVG_{2^{j+1}}(X)[t])^2}$$

$$\text{where } MS = (RMS_{2^j}(X)[t-2^j{\cdot}\Delta t])^2 + (RMS_{2^j}(X)[t])^2$$

$$\text{and } MSX = (AVG_{2^j}(X)[t-2^j{\cdot}\Delta t])^2 + (AVG_{2^j}(X)[t])^2$$

For $j \geq 1$, we use four additional types of features per sensor. The two below capture the change in the average sensor value and root mean square error over time.

$$\Delta AVG_{2^{j+1}}(X)[t] = AVG_{2^j}(X)[t] - AVG_{2^j}(X)[t-2^j{\cdot}\Delta t]$$

$$\Delta RMS_{2^{j+1}}(X)[t] = RMS_{2^j}(X)[t] - RMS_{2^j}(X)[t-2^j{\cdot}\Delta t]$$

The final two types of features are $|\Delta AVG|$ and $|\Delta RMS|$, the absolute values of the feature types defined above.

The set of features defined above can be used with a wide range of different classification methods. A good overview of various methods can be found in [Kotsiantis, 2007]. We chose Decision Trees, which have received little attention for cubicle-based occupancy detection.

5 DECISION TREE METHOD

True to its name, the *Decision Tree* classification method selects a class by descending a tree of possible decisions. At each internal node in the tree, a single feature value is compared with a single threshold value. One of the two child nodes is then selected based on the result of the comparison. When a leaf node is reached, the single class associated with that node is the final prediction. Correlations between features can be exploited, as a particular node's feature will only be used if the features of upper nodes produce a certain path through the tree. For an example of a decision tree, see Figure 4.

A decision tree is generated automatically from *training data*, sets of features coupled with known classes. As explained in [Quinlan, 1996], the feature selected for the root node of the tree is the one with the highest information gain, and the threshold is selected based on a quantity known as the *gain ratio*. For a lower node, the feature and threshold are selected in the same fashion using the subset of the training data that would reach that node. Nodes are only created if they are reached by at least K points in the training data, for some arbitrary K. The tree is also pruned according to the Minimum Description Length principle, as described in [Mehta et al., 1995].

The Decision Tree method is compelling in part because the trees themselves are intuitive and informative. Each path through a tree consists of a combination of features which tend to work together to distinguish between classes. This simplicity gives useful insights into the inner workings of the method. In contrast, the outputs from alternative methods such as SVM, Neural Networks and HMM, as used in [Lam et al., 2009a], are very difficult to interpret and respond to in an exploratory manner. Also, these methods leave feature selection to the user, whereas the Decision Tree method has this process built-in. Lastly, unlike SVM, Decision Tree analysis are not sensitive to the scale of the input data, so no conditioning of the data is necessary.

One weakness of Decision Trees is the danger of overfitting, where statistically insignificant patterns end up negatively influencing classification results. The pruning mentioned above reduces the risk of overfitting, but results in smaller decision trees that exploit fewer features.

6 EXPERIMENTAL SETUP

We used the open source application KNIME (Konstanz Information Miner) 2.1.2.0024559 [Berthold et al., 2007] as the environment to conduct our data analysis. KNIME is an application which provides a number of data mining and statistical analysis algorithms. Specifically, we used its Decision Tree implementation to conduct our study.

The data suppled to KNIME was prepared outside of the application using Python scripts which would calculate the feature values and align them based on time. We chose $\Delta t = 1$ minute; thus, for every sensor and every minute, dozens of raw samples were reduced into an average value and a noise level (the AVG_1 and RMS_1 features in Section 4). From these 1 minute values, additional features were calculated for time intervals of widths 2, 4, 8, 16, 32, and 64 minutes. The 64-minute maximum time width corresponds to $j_{max} = 6$ ($64 = 2^{j_{max}} \cdot \Delta t$). No single feature would use data from more than 64 minutes in the past.

In the end we had 7 individual sensors, 2 types of features calculated for a time width of 1 minute, and 6 types of features calculated for each of 6 longer time widths. This yielded a vector of 266 feature values for each minute of acquired data. The vectors were used in conjunction with the occupancy data to train and test Decision Trees classifiers. We limited the size of the decision trees by selecting $K = 10$.

7 RESULTS

The experiment consisted of several trial conditions, each involving different combinations of features. All conditions were evaluated using seven-fold cross validation. This means that for a given day in our data set, we used the data from that day as the validation set and used the data from the six remaining days to train the classifier. A result for each of the seven days was produced by calculating the percentage of correctly classified validation points. A single measure of accuracy was produced for the experiment by averaging all seven results. Occupancy detection accuracies are reported in Table 3.

CO_2	Current	Light	Motion	Sound	Accuracy
		×			81.019%
				×	90.787%
×	×	×		×	94.242%
×					94.679%
	×				96.267%
×	×	×	×	×	96.267%
			○		97.943%
×			×	×	98.213%
			×		98.441%

Table 3. Occupancy detection accuracies achieved using various combinations of sensor types. Each × indicates that all features of the associated sensor type were made available for the associated trial condition. The ○ indicates that only a single motion feature was used.

It is important to recognize that these classification accuracies are sensitive to the time interval Δt, the behavior of the observed occupant, and other experimental conditions. In a different setting, for example, a CO_2-based classifier may yield an accuracy significantly higher or lower than the 94.7% we obtained. However, based on these results, it is reasonable to suspect a CO_2-based classifier to outperform a light-based detector in an office setting, and a motion-based detector to outperform both. Our analysis focuses not on the overall magnitude of the classification accuracies, but rather on the relative differences in accuracy obtained for each set of conditions.

7.1 Results by Sensor Type

As a baseline measurement, we ran trials using all features of only one sensor type at a time. Light features performed the worst at determining occupancy, followed by sound, CO_2 and current. The best performing group of features of a single sensor type were those derived from motion.

We then ran trials in which the classifier was trained with combinations of features derived from multiple sensor types. By combining features of all sensor types except the best performing sensor, motion, we achieved a classification accuracy greater than that of light and sound features alone. Adding motion features to the mix yielded an accuracy greater than that of CO_2 and current features alone. Under the assumption that overhead lighting conditions and electricity consumption were leading to dubious classifications, we removed features of those sensor types. Removing light and current improved accuracy by 2%.

Surprisingly, no classifier trained with features of multiple sensor types was able to outperform the classifier trained with motion features alone. In fact, a decision tree trained with all available data from all sensors performed worse than a single-node decision tree exploiting only one motion feature. The use of a single-node decision tree is equivalent to applying a threshold to a single feature. A multi-node decision tree still produced the best results, but only when using motion features at all nodes.

Each of the experiments we described so far exhibited different characteristics on the weekend when the occupant was not present for 48 straight hours. While the classifier that was trained using only motion features gave us the best accuracy overall, it was only able to achieve 98.3% accuracy over the weekend. However, classifiers trained with either current features or sound features were able to achieve 100% classification accuracy during this period of extended absence.

7.2 Results by Feature Type

Here we attempt to quantify which features in particular proved to be the most useful in determining occupancy. The metric below computes a non-dimensional score value for each feature. High scores indicate high importance, and vice versa. The scoring formula favors features which are frequently used by decision nodes and are used at decision nodes which are close to the root of the tree.

$$Score(feature) = Occurrences(feature) \cdot \frac{1}{2^{AvgTreeDepth(feature)}}$$

For example, suppose that the feature $\Delta RMS_{64}(Sound)$ occurs in three decision trees at levels 1, 2, and 4. The score for this feature is then calculated as follows.

$$Score(\Delta RMS_{64}(Sound)) = 3 \cdot \frac{1}{2^{(1+2+4)/3}} = 0.595$$

We applied the formula to the set of seven decision trees produced for the trial condition in which all features of all sensor types were used. A summary of the top 16 scoring features is depicted in Figure 2. Overall, the feature most highly favored by the method was $RMS_2(Motion)$, as this feature was chosen as the root decision in most trees. The relative strength of this feature is further emphasized by the fact that in all trees, decisions which were children of decisions based on $RMS_2(Motion)$ very rarely chose a different classification than was already chosen by $RMS_2(Motion)$. As indicated in Table 3, this feature alone produced a classification accuracy of 97.9%. The addition of other motion features only increased the accuracy to 98.4%.

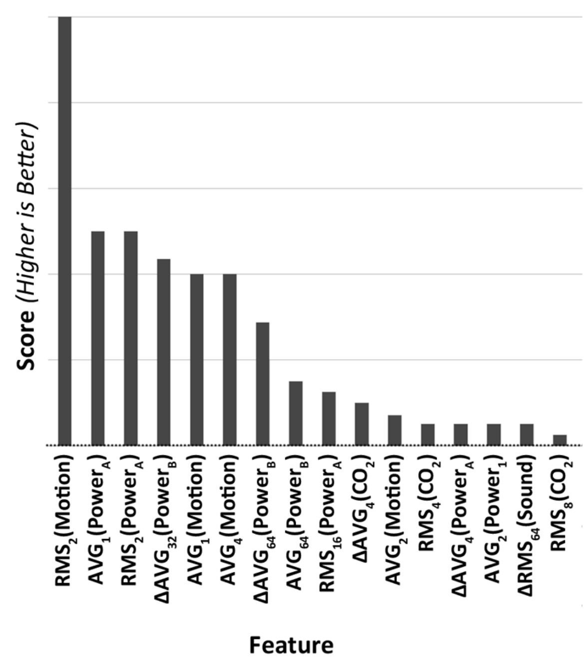

Figure 2. Simple feature scoring based on weighted feature occurrences in decision trees.

8 DISCUSSION

Simultaneously using features derived from multiple sensor types was no better than using features derived from a motion sensor alone. Our initial intuition was that the presence of additional sensor types would allow us to capture occupancy trends which a motion sensor alone is unable to detect. However, it seems that the presence of additional sensor types did

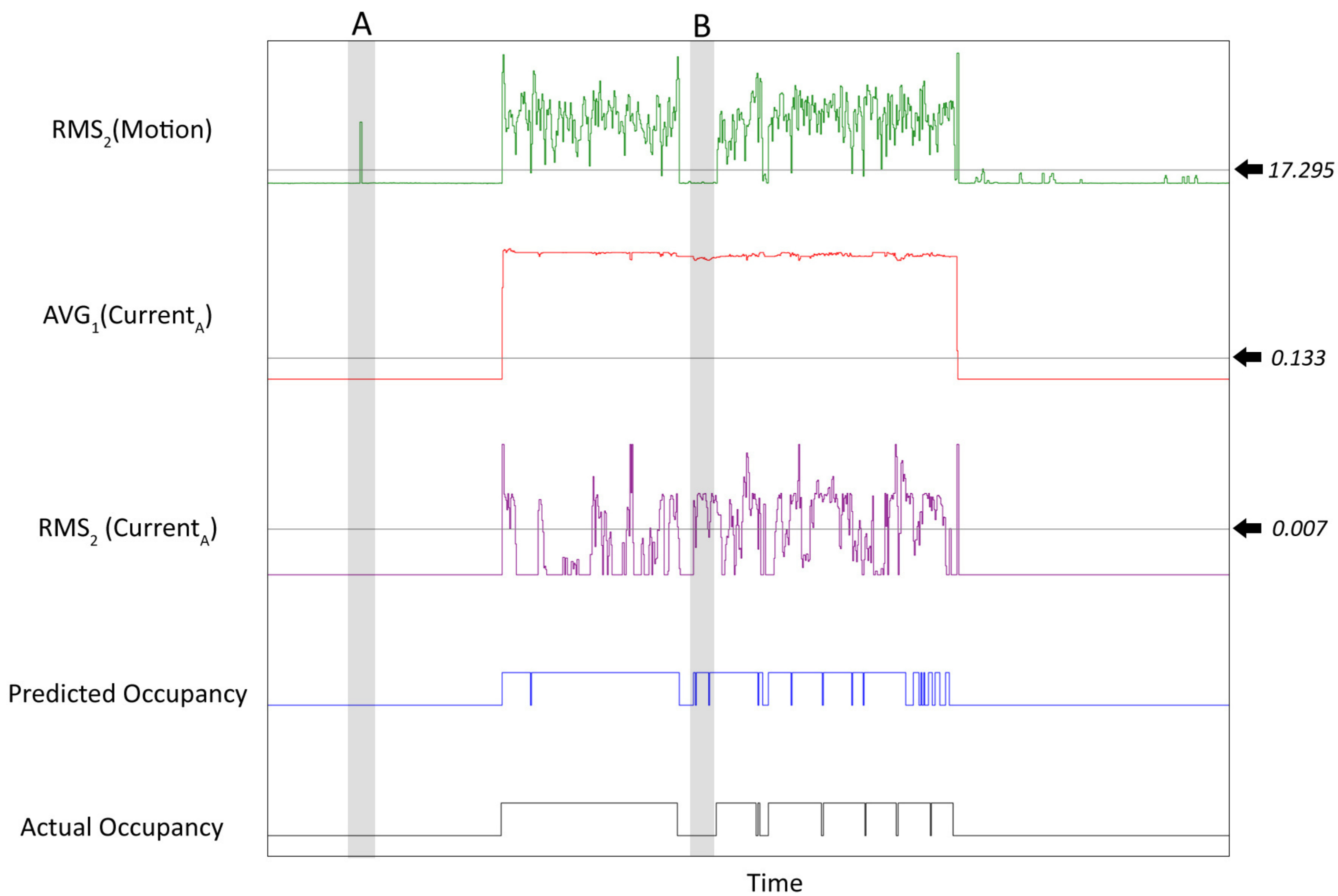

Figure 3. Plot of three features and both predicted and actual occupancy for a single day. For occupancy signals, the upper level indicates presence and the lower level indicates absence. The time periods A and B are discussed in Section 8.

as much harm as good. Here we analyze one of the classifiers to illustrate how additional sensor types sometimes improve occupancy detection, yet sometimes undermine it.

Figure 3 provides plots of the root feature of the classifier, $RMS_2(Motion)$, the two second-level features, $AVG_1(Current_A)$ and $RMS_2(Current_A)$, the predicted occupancy, and the actual occupancy. The predicted occupancy was produced by the decision tree trained using all features of all sensor types. The tree itself is shown in Figure 4.

Over time period A, $RMS_2(Motion)$ produces a spike that crosses the threshold 17.295 when the occupant is not present. In the absence of additional features, this would result in a classification error. However, since $AVG_1(Current_A)$ is below its own threshold of 0.133 over the same time period, it correctly classifies the occupant as being absent. The additional sensor type, in this case a current meter on the computer monitor, improves the classification based on motion data alone.

Over time period B in Figure 3, we can see that $RMS_2(Motion)$ is well below the threshold 17.295, correctly classifying the occupant as absent. However, when we de-

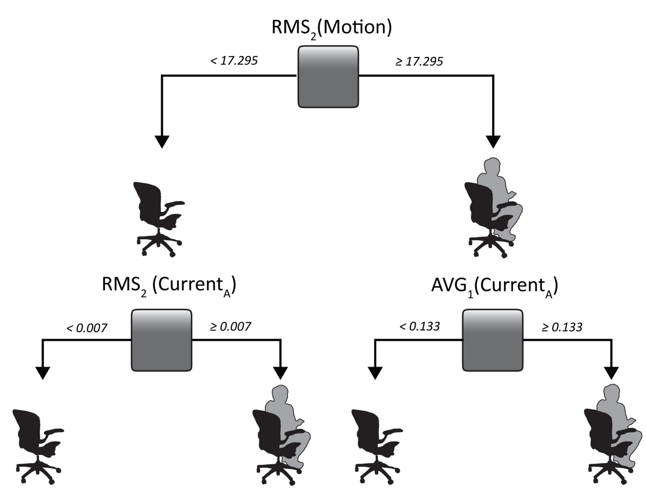

Figure 4. Top two levels of a decision tree trained using all features of all sensor types.

scend one level in the tree, $RMS_2(Current_A)$ undoes the correct classification produced by motion. Here the presence of an additional sensor type leads to a misclassification that would not have otherwise occurred. This is an example of how overfitting may occur in a decision tree, where a low-level node exploits a misleading pattern in the training data.

In our experiment, cases like that of time period B outweighed cases like that of time period A. In some instances the additional sensors of various types corrected decisions based on the motion sensor alone. But surprisingly, in a greater number of instances the motion-based classification ought not to have been changed.

Classification errors can be loosely classified into two categories: *transition errors* and *spurious errors*. Transition errors occur over time periods when the occupant has just arrived or has just left their workspace. When these types of errors occur the classifier is typically late to predict the transition change by an average of one to two minutes. This type of error is largely due to the fact that we only use past data to inform our classifications, generally making the predictor unable to output a state change until past information of the transition is available. Transition errors may also be introduced by slight inaccuracies in the ground truth occupancy data. For instance, the video observer may have marked the occupant absent before they were entirely out of sensor range.

We describe all other classification errors as spurious. Spurious errors can be credited to a multitude of factors including limited training data, inadequate selection of features, overfitting, or anomalies in sensor values or occupancy data.

We found that the majority of classification errors were transition errors. A close examination of Figure 3 reveals several transition errors where the vertical edges of the two occupancy plots do not perfectly align. There are also a few noticeable spurious errors in the plots, the first appearing as a brief predicted absence about halfway between time periods A and B. In building automation applications, relatively short misclassification errors of either type may be tolerable. However, spurious errors are less tolerable when collecting data to calibrate simulations and perform real-time monitoring.

One should be aware that the make and model of sensors used for training must be the same as what is used for classification. To illustrate why, consider a motion sensor that outputs a non-standard non-dimensional value. In this case, one make of motion sensor is not necessarily able to use the decision thresholds arrived at via training with a different motion sensor. Even sensors which output dimensional values such as current (Amperes) and sound (Decibels) are subject to the same limitations, as sensing ranges, resolutions, and sampling frequencies vary by sensor make and model.

9 CONCLUSION

Having improved occupant detection accuracy from 97.9% to 98.4% in our study, Decision Trees may well outperform simple thresholds when applied to multiple features derived from a single motion sensor. Interestingly, the inclusion of other types of sensor data did not improve overall accuracy when combined using a decision tree. The implication is that, while Decision Trees may improve occupancy detection systems based on motion sensors alone, one risks overfitting if multiple types of sensors are combined.

It remains intuitive that the use of other types of sensors in conjunction with motions sensors could improve an occupant detection system. However, in light of our results, it might be best to explore alternative classification methods known to be less prone to overfitting.

Despite mixed results in terms of classification accuracy, we demonstrated how Decision Trees can be used to explore the relationship between sensor types, features, and occupancy. The method is compelling in large part because one can understand how each sample was classified, and often determine the cause of a classification error.

If deployed, our occupancy detection method might have to be adjusted to suit the particular application. The objective of our study was to detect present occupancy at a one-minute resolution, which could be useful for visualization and security applications. For a building control system, one might modify the method to predict occupancy between the present time and an hour or two in the future. In some applications, the real-time aspect of our classifier may not be priority. For example, if one is collecting occupancy data to calibrate a simulation of occupant behavior, then it should be reasonable to use future sensor data as well as past information to produce the feature values.

Future work will need to focus on capturing data from not only different individuals, but individuals who exhibit considerably different behavior in a wide variety of settings. For example, an occupant who is primarily a computer user would produce different patterns in the sensor data and occupancy profiles that one who spends more time on the phone. Data from a diverse set of occupants may yield a generic classifier that is able to detect the presence and absence of many different individuals.

10 ACKNOWLEDGMENTS

Thanks to our colleagues Michael Glueck, Conan Huang, Michael Lee, Wei Li and Alex Tessier for their invaluable support. Also thanks to John King, Regional Facilities Manager and Zulfikar Texiwala, Building Operator, at Autodesk Office in Toronto for their continuous support throughout our ongoing research.

REFERENCES

BERTHOLD, M. R., CEBRON, N., DILL, F., GABRIEL, T. R., KÖTTER, T., MEINL, T., OHL, P., SIEB, C., THIEL, K. AND WISWEDEL, B., 2007. KNIME: The Konstanz Information Miner. In Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007). Springer.

DODIER, R. H., HENZE, G. P., TILLER, D. K. AND GUO, X., 2006. Building occupancy detection through sensor belief networks. *Energy and Buildings*, volume 38, 9, 1033–1043.

HAILEMARIAM, E., GLUECK, M., ATTAR, R., TESSIER, A., MCCRAE, J. AND KHAN, A., 2010. Toward a Unified Representation System of Performance-Related Data. In the 6th IBPSA Canada Conference, 117–124.

KOTSIANTIS, S. B., 2007. Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, volume 31, 249–268.

LAM, K. P., HÖYNCK, M., DONG, B., ANDREWS, B., CHIOU, Y.-S., ZHANG, R., BENITEZ, D. AND CHOI, J., 2009a. Occupancy Detection Through an Extensive Environmental Sensor Network in an Open-Plan Office Building. In Proceedings of the 11th International IBPSA Conference, 1452–1459. Glasgow, Scotland.

LAM, K. P., HÖYNCK, M., ZHANG, R., ANDREWS, B., CHIOU, Y.-S., DONG, B. AND BENITEZ, D., 2009b. Information-Theoretic Environmental Features Selection for Occupancy Detection in Open Offices. In Proceedings of the 11th International IBPSA Conference, 1452–1459. Glasgow, Scotland.

MEHTA, M., RISSANEN, J. AND AGRAWAL, R., 1995. MDL-based Decision Tree Pruning. In Proceedings of the First International Conference on Knowledge Discovery, 216–221. Montreal, Canada.

MENDLER, S., ODELL, W. AND LAZARUS, M. A., 2006. The Guidebook to Sustainable Design. John Wiley & Sons Press, second edition.

PADMANABH, K., V, A. M., SEN, S., KATRU, S. P., KUMAR, A., C, S. P., VUPPALA, S. K. AND PAUL, S., 2009. *i*Sense: A Wireless Sensor Network Based Conference Room Management System. In BuildSys '09: Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, 37–42. Berkeley, CA, USA.

QUINLAN, J. R., 1996. Improved Use of Continuous Attributes in C4.5. *Journal of Artificial Intelligence*, volume 4, 77–90.

RAMOSER, H., SCHLÖGL, T., BELEZNAI, C., WINTER, M. AND BISCHOF, H., 2003. H.bischof. shape-based detection of humans for video surveillance. In In Proc. of IEEE Int. Conf. on Image Processing, 1013–1016.

TARZIA, S. P., DICK, R. P., DINDA, P. A. AND MEMIK, G., 2009. Sonar-based measurement of user presence and attention. In Ubicomp '09: Proceedings of the 11th international conference on Ubiquitous computing, 89–92. ACM, New York, NY, USA.

YILMAZ, Z., 2007. Evaluation of energy efficient design strategies for different climatic zones: Comparison of thermal performance of buildings in temperate-humid and hot-dry climate. *Energy and Buildings*, volume 39, 3, 306–316.