Indoor Positioning System for Smart Homes based on Decision Trees and Passive RFID

Frédéric Bergeron*, Kevin Bouchard, Sébastien Gaboury, Sylvain Giroux, and Bruno Bouchard

Université de Sherbrooke, Sherbrooke, Canada
Université du Québec à Chicoutimi, Saguenay, Canada
{frederic.bergeron2,kevin.bouchard,sylvain.giroux}@usherbrooke.ca,
{sebastien.gaboury,bruno.bouchard}@uqac.ca

Abstract. This paper presents a novel Indoor Positioning System (IPS) for objects of daily life equipped with passive RFID tags. The goal is to provide a simple to use, yet accurate, qualitative IPS for housing enhanced with technology (sensors, effectors, etc.). With such a service, the housing, namely called smart home, could enable a wide range of services by being able to better understand the context and the current progression of activities of daily living. The paper shows that classical data mining techniques can be applied to raw data from RFID readers and passive tags. In particular, it explains how we built several datasets using a tagged object in a real smart home infrastructure. Our method was proven very effective as most algorithms result in high accuracy for the majority of the smart home.

Keywords: RFID; indoor positioning; smart home; decision tree; random forest

1 Introduction

Many occidental countries are faced with the ageing of their population, causing an explosion in health related costs [16]. Indeed, ageing is associated with a higher prevalence of many illnesses like Parkinson's disease, Alzheimer's disease, etc. Persons afflicted by these diseases soon require constant care often leading to institutionalization. In addition to resulting in high costs for the society, institutionalization might significantly decrease social interaction and reduce autonomy which account for a lower quality of life [4]. One way to reduce the impact of these problems is to enable aging in place. Smart homes are being developed by many research teams around the world to address this challenge. One of the primary difficulty faced by researchers lies in the monitoring the progress of ongoing activities of daily living (ADLs), formely known as the activity recognition problem [6]. Over the years, numerous algorithms have been designed for that purpose [14], but most of them do not distinguish the individual steps of an

* We would like to thanks our main sponsor, INTER

activity. The main explanation for this limit is that information acquired from sensors data is very noisy and limited.

Information of a spatio-temporal nature could greatly help improve the recognition of ADLs. A smart home contains various technologies that could provide spatio-temporal information. Some research teams have designed Indoor Positioning Systems (IPSs) using microphones to analyse sounds, while others have used low resolution cameras (often considered as very intrusive for the resident) or even communication protocols such as Bluetooth or Wi-Fi to track objects. In this work, we investigated the exploitation of passive Radio Frequency IDentification (RFID) tags because they are inexpensive, small and resistant. They are also the only technology to be small enough the be carried on in an unobtrusive way. On the other hand, they are also very inaccurate and subject to interferences from many sources, causing great difficulties for the design of an IPS. RFID has been used before in the literature for indoor positioning [8]. However, most systems focus primarily on precision, while high accuracy and deployment simplicity are more important.

In this paper, we propose a new Indoor Positioning System for smart home based on passive RFID technology and decision trees. The goal is to enable real-time tracking of the objects used in daily life activities. This new solution palliates the problem of the lack of accuracy of the previous studies by proposing a qualitative solution adapted to the precision of the RFID system. The method is said to be qualitative since it provides a logical position (defined by a qualitative zone) instead of precise coordinates. Our goal was to design an easy-to-deploy solution that would be extensible to the addition of antennas and to the modification of an RFID system. The datasets collected for these tests are available to the scientific community on www.Kevin-Bouchard.com.

2 Related work

In this section, we present major works that addressed the problem of indoor localization in the last few years. There are many applications for IPSs. For example, in robotics, localization is important for action planning. A number of research has been done on the subject, and yet, good positioning and reliable tracking with passive RFID remains hard to achieve [8]. Many researchers have explored other ways to build an IPS. Some of them are described in the next sub-sections.

2.1 Passive RFID Localization

There are two types of RFID tags: actives and passives. Active tags have an inner energy supply that allows them to emit a strong signal at any time. Passive tags have no inner energy supply. On the contrary, they use the energy contained in an inbound signal to activate themselves and send back a signal with the remnant energy. While having a much longer life, passive tags have a shorter range.

Most RFID systems are based on LANDMARC [13]. This system was the first to use reference tags placed at strategic locations to achieve good precision on positioning. With four readers and one reference tag per square meter, they reached a mean error of one meter. Joho et al. [10] presented a model with a localization error of about 35cm. They achieved that by using received signal strength information (RSSI), antenna orientation and references tags with a mobile reader that gathers information to compare with the real position.

2.2 Trilateration, triangulation

In addition to the various declinations of the LANDMARC system, many researchers have worked on the classical algorithms of trilateration (distances) and triangulation (angles). These methods are often ignored in context where incoming data is noisy and inaccurate, but some researchers still obtained very interesting results. In particular, Chen et al. [5] exploited ZigBee (a radio-frequency technology used for communication between objects in a small environment) to perform trilateration. They implemented a fuzzy inference engine composed of only one variable. Their engine correlates the RSSI of a transmitter to localize to the distance separating it from a receiver. Fortin-Simard et al. [8] also worked on trilateration. They exploited multiple filters to preprocess the RSSI as a preliminary step to an elliptical trilateration. Their method achieved a localization of up to 15 cm.

The main problem of these approaches is that they focus on increasing the precision of the localization at the expense of the accuracy over time. As our goal is to obtain an accurate tracking system for activity recognition, these methods are not well-adapted. Also, these systems are usually complex to implement and require a lot of technical expertise.

2.3 Positioning with data mining

To palliate the limitation of the literature, learning algorithms could be exploited. S.L. Ting et al. [17] studied the possibility of using passive tags to identify qualitative positions. They placed an antenna at each corner of a room and divided the area into nine zones of one square meter. They then recorded 6 readings for each zone and used the average of these readings to form a look-up table. They then used the Euclidian distance to find the nearest neighbor to the RSSI of the detected object whose position is to be determined. Their overall accuracy is 93% over 90 trials. We extend on this work by using a much more zones we many algorithms.

Decision trees have already been used for the purpose of localization, along with nearest neighbors and Bayesian methods. They are often called fingerprints methods. In a series of papers, J. Yim [19] makes an extensive use of those methods to build IPSs using wireless local area network (WLAN). In [19], he showed that decision trees have a similar performance to a Bayesian network and a 1-nearest neighbor with WLAN.

3 Methodology

In this section, we explain in detail how we created our datasets to implement our new solution. These datasets are composed of RFID readings for each qualitative zone of the positioning system. But before exploring the details, it is important to describe the intelligent environment that served this project.

This project took place at the DOMUS laboratory of the Universit de Sherbrooke. It is a complete and realistic apartment equipped with more than a hundred sensors distributed in six different rooms: one bedroom, an entry hall, a kitchen, a dining room, a living room and one bathroom. There are twenty RFID antennas disposed to cover the whole surface of the smart home.

The Fig. 1 shows an aerial view of the smart home. As you can see, there are more antennas in the kitchen than in any other room since this is where most of the activities of daily living occur.

3.1 Creating a target object

The first step of our project consisted of creating a special object that would allow us to get good RFID readings. This object needed to be something common we could easily find in any house. We chose an empty rigid plastic bottle of water. Then, we had to select the tags to install on the object. To do so, it is important to check tags to ensure they give very similar RSSI when placed side by side. We decided to use four tags to reduce the bad angle of arrival problem. We put those tags on the bottle making sure that each tag was facing a different direction. When collecting data from the bottle, we merged readings for the four tags to keep only the highest reading from each antenna. This gave us an improved special tag that covered each direction, always facing the nearest antennas. A complete reading of our special tag is a vector of twenty negative integers representing the measured signal strength in decibel. Values range from minus thirty to minus seventy, as allowed by the configuration module. We put the tags on an object because for the final application we intent to track only the objects in the housing and not directly the inhabitants.

3.2 Creating zones

The second step of this project consisted in defining qualitative zones. At first, we had no way to know what kind of precision we would be able to reach with our setup other than estimating. Therefore, we started by dividing the dining room into zones of 100cm X 100cm. We evaluated the performance by training a basic decision tree with the SimpleCART algorithm from the Weka tool [9], using cross-validation. Next, we tried with 75cm X 75cm, 60cm X 60cm, 45cm X 45cm, 30cm X 30cm and finally 20cm X 20cm. We stopped at twenty because it was near the size of the bottle and more precision would not help since the error rate was increasing fast. Additionally, the most precise non qualitative positioning algorithms in the literature report precision around 20cm [8]. From this preliminary step, we collected many datasets for that single room that were

not used for the final design of our solution, but could be useful in the future to improve accuracy if the positioning precision proved to be less important for our final system.

Once we established how precise the system could be, we were able the define zones for each room. Fig. 1 presents a map of all the defined zones. Clearly, not all of the zones are the same size, because not all rooms require the same precision for our final goal of recognizing activity of daily living. The figure also shows that zones located on the edge of each room are bigger because the precision is less important since they cover places that a person would not usually go. The precision in the bathroom and in the bedroom is sixty centimeters. The kitchen and the dining room have a precision of twenty centimeters and the other rooms have seventy five centimeters. The counter in the bathroom has a higher precision of thirty centimeters since more complex activities can happen there (e.g.: brushing teeth, shaving, etc.). This causes the surrounding zones to have heterogeneous shapes.

3.3 Collecting datasets

The third step of the project, once we had a special object and all our zones, was to collect data from them. We collected a single dataset for each room in the DOMUS' smart home, each one identified by the name of the room. We placed our special bottle on a bench at the same height as the antennas and collected exactly fifty readings for each zone. Antennas were calibrated to send a signal every 750 milliseconds. This allowed us to reduce to a minimum the risk of interference between antennas as the probability of an antenna to communicate at the same time as another was very low. The bottle was always placed in the middle of each zone to ensure the regularity of the readings. For the datasets, we did not collect any readings close to a zone border.

After several days of data collection, we ended up with a datasets per room. They contain 47,989 readings from 963 qualitative zones. Moreover, every reading was labeled to its zone, making the datasets perfect for supervised learning. We decided to make these datasets available for other smart home researchers on social networks and on our personal websites. The reason we decided to do so is that it is very difficult to repeat experiments in the context of smart home research to validate and improve research. Moreover, many researchers in our field do not have access to RFID technology and they might want to experiment with it before investing a considerable amount of money to equip their infrastructures.

4 Experiments and Results

An ADL recognition algorithm like the one we planned to design needs a real-time tracking system. A real-time tracking system needs a fast and accurate positioning system. This section presents the algorithms that we used and the accuracy they gave on each dataset. We also briefly describe each family of

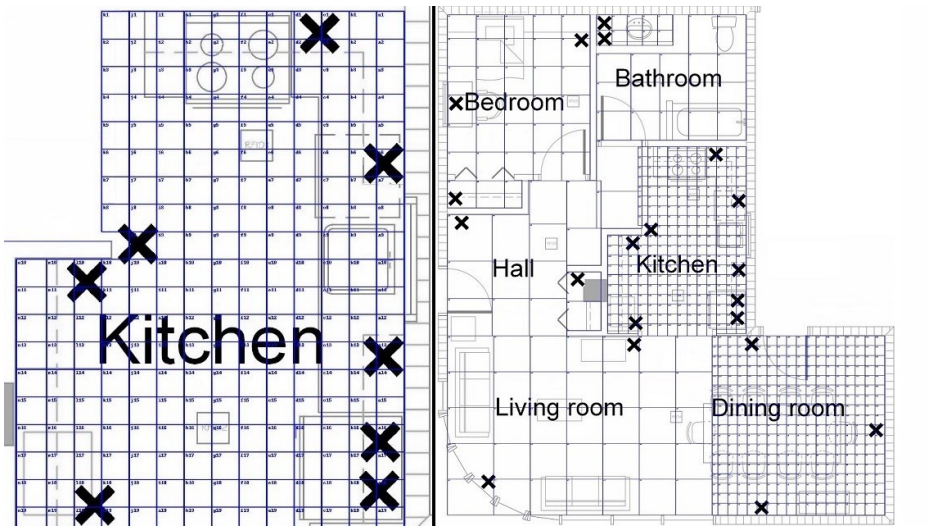

Fig. 1. On the left, a close view of the kitchen map. On the right, the overall map of the DOMUS smart home. The RFID antennas are marked by a X. The grid represents the scaled qualitative zones

Algorithms used. We conclude the section with a simple algorithm that allowed us to merge qualitative zones under certain conditions to improve accuracy.

As mentioned previously, our goal was to determine if we could use known data mining algorithms to create an indoor positioning system using passive RFID. To do so, we tested several algorithms, mostly trees, on each collected dataset. All algorithms are the implementations available in Weka [9] and we used them with default parameters. Table 1 shows an overview of the results we obtained. Accuracy was computed using 10-fold cross-validation, as set by default in Weka. We did this in order to do a maximal number of tests with the widest range of algorithms before searching for the best hyper-parameters. Also, we were looking to develop a simple positioning system that could perform well in almost any given indoor environment without having to spend great effort in fine-tuning the algorithm every time. Thus, using default parameters allowed us to rapidly find the algorithm most suited for positioning in each room. As a side note, it is important to mention that our system supposes the existence of another positioning system able to determine in which room each object is located.

We can see in Table 1 that, as expected, almost all algorithms had a better accuracy for rooms with bigger zones. The size of the zones not only allows for more variation in received signal strength, but also having less target classes makes it easier for most algorithms to discriminate between them. In fact, there are a finite number of RSSI vectors we can obtain in our apartment. The maximum number would be 2040 (40 is the possible variation in decibels and 20 the number of antennas), but in fact the number is much smaller due to the

boundary of the environment. Many vectors are just impossible to get as they would be outside, in walls, or because they were out of the antennas' reach.

Table 1. Accuracy for different learning algorithms on all datasets

Dataset	Accuracy								
	BNet	NNet	1-NN	Cart	J48	RT	NBT	LMT	RF
Hall	96,375	51,125	94,875	95,625	96,875	95,625	96,375	95,500	97,750
Living Room	97,657	91,257	94,800	92,400	92,057	89,314	94,686	93,314	97,600
Kitchen	84,983	75,639	78,824	73,403	74,966	67,815	83,227	74,908	88,916
Dinning Room	74,931	65,205	73,251	72,492	73,375	71,084	73,578	70,392	76,727
Bedroom	95,697	86,485	93,212	91,515	92,849	89,818	92,485	92,424	96,485
Bathroom	95,111	87,482	90,370	90,266	91,630	88,148	90,963	92,593	96,074
Average	90.792	76.199	87.555	85.950	86.959	83.634	88.552	86.522	92.259

4.1 Trees

First, we will present some trees we used. We trained some very classical trees, but we also trained more exotic ones to see if they would perform well with our data. We have explained them in the next few paragraphs.

Decision trees The first type of trees we present are the decision trees. They are simple yet efficient on our data. J48 is the name given by Weka to its implementation of the C4.5 algorithm [15]. C4.5 uses the information gain metric to find the best attribute to split on in each node until there is only one class left or a minimum number of examples is reached. Then a pruning step is done trying to remove leaves that do not bring any accuracy gain. Next, we tested SimpleCart [3], Cart in Table 1, which is another decision tree. It differs from C4.5 at the pruning step where it uses the cost complexity pruning. There was not any conclusive gain with SimpleCart over C4.5, which suggests that pruning does not yield significant differences with our datasets.

The last simple tree we present is the Random Tree. It is simply a tree where only a subset of attributes is considered for each split. It is therefore well adapted to datasets consisting of a large number of attributes. The size of this subset is $\log_2(D)+1$, where D is the number of attributes. There is no pruning. It performed slightly worse than the other trees. Indeed, our datasets are composed of only 20 numerical attributes and Random Tree is better suited for contexts with very high dimensionality.

Decision Trees with a Model on Leaves We just saw three different trees that work under the same basic principles. This sub-section presents two other trees that have a model on their leaves or nodes.

NBTree [11] is a tree with the main characteristic that there is naive Bayes classifiers on the leaves. The author of this model affirms that it can outperform both the normal tree and the naive Bayes classifier, especially on large datasets. His affirmation proved to be true in our experiment. NBTree was better or equivalent to C4.5 on most datasets, while being significantly better in the kitchen. As our final goal is to build a system for ADL recognition, it makes this model a particularity good choice.

Logistic Model Tree [12], LMT in Table 1, is a tree whit a logistic regression function at the leaves. It produces big trees, thus being longer to train than other with tried for similar accuracy. Indeed, a bigger tree implies more leaves and more logistic functions to learn.

Trees forest The last tree algorithm that we trained was the Random Forest [2]. Random Forest is simply 100 Random Trees trained separately, exactly the same way as described before. The predicted class is then the modal class between all 100 predicted classes. This forest surpassed the single random tree, thus being by far the most accurate model on our datasets. It even outperformed the Bayesian Network while being shorter to train and faster to use. Nevertheless, it is slower than basic classifiers such as C4.5 and SimpleCART.

4.2 Other Algorithms

In the previous section we discussed the main decision trees algorithms that we decided to test in this work. This section focuses on other algorithm families we tried, the nearest neighbors, the Bayesian Network and the Multilayer Perceptron.

Nearest Neighbors Alone in his family, the k-nearest neighbors (k-nn) algorithm [1] is a very simple method that consists of finding the k closest known examples to a given unclassified example and predicting the modal class among those examples. There are many distance functions that can be used by k-nn. As we are in a continuous 2-dimentional context, we chose to use the Euclidian distance, which is also the default distance in Weka.

We tried this algorithm with k values ranging from 1 to 5. Table 3 shows the accuracy obtained on the hall dataset. It shows that the best value for k is one. We observed the same results for all datasets, where the accuracy always diminishes as k increase. However, the difference between k equals one and k equals five could be as big as eight percent. Still, in most cases, the difference was not that big, varying of about one percent per k. The main reason behind these results is that a bigger k makes boundaries between classes less distinct. In future work, it would be interesting to verify if the k-nn algorithm performs better when coupled to a heuristic to select a good k.

Bayesian Network Next, we trained a Bayesian network. It is a probabilistic model presenting itself as a directed acyclic graph that we can use to represent

Table 2. Accuracy for different K

Dataset	Accuracy for K				
	1	2	3	4	5
Hall	94,875	94,625	92,375	91,75	91,000
Living Room	94,800	93,600	92,800	92,514	91,429
Kitchen	78,824	74,706	74,261	72,605	71,319
Dinning Room	73.251	71.318	70.979	70.164	69.787
Bedroom	93,212	92,364	92,424	91,636	91,455
Bathroom	90,370	88,667	88,444	87,482	86,074
Average	87.555	85.880	85.214	84.359	83.511

a probability distribution of classes over attributes. It only works with discrete values, so our data were discretized before the training. Training a Bayesian network can be seen as two separate steps: learning the network structure and learning the probability tables. The one present in Weka offers numerous possibilities in the choice of algorithm for each of those steps. We used the K2 algorithm to learn the network structure. K2 is a hill climbing method that uses a fixed ordering of variables to maximize quality measure of the network structure. In our case, the quality measure was the Bayesian metric from Cooper & al. [7] (see equation (1)), a measure that tends to approximate the likelihood of the graph. The graph was initialized as a Naive Bayes Network. This means that the classifier node is connected to all other nodes. In equation 1, BS represents the network structure of the database D. $P(B_S)$ is then the prior network structure and r_i is the cardinality of the data. N_{ijk} is the number of cases in D where the variable x_i as the value v_{ik}.

$$Q_{K2}(B_S, D) = P(B_S) \prod_{i=0}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(r_i - 1 + N_{ij})!} \prod_{k=1}^{r_i} N_{ijk}! \tag{1}$$

To learn the probability tables, we used the SimpleEstimator. It estimates the conditional probabilities directly using the given data. A smoothing constant of 0.5 is used by default when computing the probability tables.

This model proved to be one of the best we have trained, only matched by the Random Forest. However, it requires a very long training, especially on the bigger datasets like the kitchen. Indeed, the complexity of complete inference is NP [18]. Classification could be longer as well because it requires a lot more computation than decision trees which are usually under $O(\log n)$. In future work, it would be interesting to verify if a Bayesian Network is usable to perform real time classification for an indoor positioning system.

Multilayer Perceptron The Multilayer Perceptron consists of many non-linear nodes linked together by weighted links. In Weka, all nodes are a sigmoid function. Weights and biases in the network are trained using the back-propagation algorithm that allows an efficient transmission of the gradient throughout all nodes. This gradient comes from the loss function that the learning phase

tries to minimize. A learning rate of 0.3 is used to monitor the learning, along with a momentum of 0.2. Each training example was seen 500 times by the network. There was only one hidden layer of size $D + C$, where D is the number of attributes and C the number of classes to predict.

Using only the default parameters, the Multilayer Perceptron was very inconsistent over our different datasets. Its accuracy was at most satisfactory and was often worse than all other algorithms. However, we believe that with appropriate fine-tuning it could be one of the best algorithms. Still, it was not in the scope our experiment to fine-tune the learning algorithms and it would have been biased to adjust this algorithm when the other algorithms could also have been fine-tuned. We tested the multilayer perceptron to see if it could be exploited as easily as simple decision trees. As we expected, it requires a bit more effort to use it. We also think that it would perform better with a larger dataset of a higher dimensionality.

4.3 Merging Qualitative Zones

One of the problems we discovered after running the algorithms mentioned above was that they all suffer from the high number of qualitative zones they have to classify. We tried to diminish the impact of this problem by merging automatically certain zones that were hardly distinguishable. In order to achieve this, we designed a simple statistical method. The general idea is presented below:

1. Train a classifier;
2. Record all erroneous classification on test set;
3. For each error, if the predicted zone is neighboring the target zone, merge the zones.

By merging only neighboring zones, we ensured that the accuracy would not drop but only improve since there would be at least one more example that would be correctly classified. Moreover, we thought that those small local merges would not affect our ADL recognition algorithms due to the small number of them. We tested the effects of the merging on CART algorithm and the accuracy improving from 1% to 7% depending on the room.

5 Discussion

In the previous section, we explained and tested several algorithms. We saw that they all perform relatively well on most datasets. Still, some are faster in making a decision and thus they should be prioritized. Indeed, decision time is an important factor for a tracking system like the one we intend to do. Neural networks could work, but they need much bigger datasets with more attributes to perform at their best. For all those reasons, we consider the Random Forest as the best algorithm to fit our needs. Not only is it one of the best in terms of accuracy, it is also fast to execute. Adding more tree does not really affect

the decision speed, as it stays under $\log_2 n$, where n is the number of classes. Another argument in favor of the Random Forest is that we can get more than one prediction. In this paper, we used only the modal prediction. But, for the tracking system, we could use the zones with the most votes and then pick the one that is closest to the previous position.

For the kitchen and the dining room dataset, we did not get as high an accuracy as the other rooms. We could solve this problem by exploiting bigger zones. In fact, we tested datasets composed of zones of 30cm X 30cm, 45cm X 45cm, 60cm X 60cm, 75cm X 75cm and 100cm X 100cm in the dining room and the accuracy climbed to respectively 78.98%, 81.78%, 88.61%, 95.04% and 96.07% with C4.5 from the 73.35% with the zones of 20cm X 20cm. However, we believe that the accuracy will greatly increase when the models will be exploited in a complete real-time tracking system by adding filtering on the incoming positions. Indeed, the reason why other methods such as trilateration work so well is because they use many filters on a high number of readings [8]. The more readings we can get, the better the algorithm will be.

6 Conclusion

In this paper, we presented a novel method for the indoor positioning of objects used in daily life in a smart home. We have shown that traditional data mining algorithms can be effectively exploited for qualitative indoor positioning. Trees are very simple algorithms that can be very accurate on this task. Moreover, they are fast to train and even faster to use, which is important when the goal is to do real-time positioning. Our qualitative approach is also adaptable, allowing us to decide the size of each zone to reflect the precision needed in each room or even for a particular sector of a room. Moreover, like trilateration techniques, data mining algorithms can use the raw received signal strength. However, they do not need a data rate as high as the former because there is no filtering. We also showed that we could merge some neighboring zones to increase the accuracy of our system. Another important contribution of this work is the twenty datasets composed of more than 47 thousands lines that are available to other researchers through our website (Kevin-Bouchard.com). Experimenting on realistic datasets is often difficult for smart home researchers due to the time required to perform the data collection and the cost of the infrastructure.

The main drawback of our method is that we need to construct datasets by manually collecting readings for each qualitative zone. While it is an easy task that does not require a particular human expertise, it is also a task that is time consuming. In future work, we aim to address this issue by automating this data collection phase in order to reduce the human involvement and decrease the length of the learning phase. For example, the special object could be placed in each zone with laser meter, the controller program could be on a tablet, a robot could place the object, etc. The next step of this project will be to test this indoor positioning system for real time and continuous tracking of objects within the DOMUS smart home.

References

1. D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine learning*, 6(1):37–66, 1991.
2. L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
3. L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.
4. M. Brown and D. Vandergoot. Quality of life for individuals with traumatic brain injury: comparison with others living in the community. *The Journal of head trauma rehabilitation*, 13(4):1–23, 1998.
5. C.-Y. Chen, J.-P. Yang, G.-J. Tseng, Y.-H. Wu, R.-C. Hwang, et al. An indoor positioning technique based on fuzzy logic. In *Vol 2 in Proceedings of the International MultiConference of Engineers and Computer Scientists*, 2010.
6. L. Chen, C. D. Nugent, and H. Wang. A knowledge-driven approach to activity recognition in smart homes. *Knowledge and Data Engineering, IEEE Transactions on*, 24(6):961–974, 2012.
7. G. F. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine learning*, 9(4):309–347, 1992.
8. D. Fortin-Simard, K. Bouchard, S. Gaboury, B. Bouchard, and A. Bouzouane. Accurate passive rfid localization system for smart homes. In *Networked Embedded Systems for Every Application (NESEA), 2012 IEEE 3rd International Conference on*, pages 1–8. IEEE, 2012.
9. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.
10. D. Joho, C. Plagemann, and W. Burgard. Modeling rfid signal strength and tag detection for localization and mapping. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3160–3165. IEEE, 2009.
11. R. Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *KDD*, pages 202–207. Citeseer, 1996.
12. N. Landwehr, M. Hall, and E. Frank. Logistic model trees. *Machine Learning*, 59(1-2):161–205, 2005.
13. L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil. Landmarc: indoor location sensing using active rfid. *Wireless networks*, 10(6):701–710, 2004.
14. P. Palmes, H. K. Pung, T. Gu, W. Xue, and S. Chen. Object relevance weight pattern mining for activity recognition and segmentation. *Pervasive and Mobile Computing*, 6(1):43–57, 2010.
15. J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.
16. H. Rafalimanana and M. Lai. World population ageing 2013. new york: United nations, department of economic and social affairs. *Population Division*, 2013.
17. S. Ting, S. K. Kwok, A. H. Tsang, and G. T. Ho. The study on using passive rfid tags for indoor positioning. *International journal of engineering business management*, 3(1.):9–15, 2011.
18. D. Wu and C. Butz. On the complexity of probabilistic inference in singly connected bayesian networks. In *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, pages 581–590. Springer, 2005.
19. J. Yim. Introducing a decision tree-based indoor positioning technique. *Expert Systems with Applications*, 34(2):1296–1302, 2008.